

ISO TC184/SC4/WG2 N 483

Date: 2001-1-31

ISO/CD 13584-25

Parts library: Logical resources: Logical model of supplier library with aggregate values and explicit content.

COPYRIGHT NOTICE:

This ISO document is a committee draft and is copyright protected by ISO. While the reproduction of committee drafts in any form for use by Participants in the ISO standards development process is permitted without prior permission from ISO, neither this document nor any extract from it may be reproduced, stored or transmitted in any form for any other purpose without prior written permission from ISO.

Requests for permission to reproduce this document for the purposes of selling it should be addressed as shown below (via the ISO TC 184/SC4 Secretariat's member body) or to ISO's member body in the country of the requestor.

Copyright Manager

ANSI

11 West 42nd Street

New York, New York 10036

USA

phone: +1-212-642-4900

fax: +1-212-398-0023

Reproduction for sales purposes may be subject to royalty payments or a licensing agreement.

Violators may be prosecuted.

ABSTRACT:

This part of ISO 13584 provides generic EXPRESS resource constructs that support the description of different kinds of aggregate data types and instance values. These resource constructs were developed as a joint effort of ISO TC184/SC4/WG2 and IEC SC3D.

This part of ISO 13584 also contains an integrated EXPRESS information model for representing supplier libraries for the purpose of exchange. This integrated information model integrates the above resource constructs with other EXPRESS resources from different parts of ISO 13884 and ISO 10303 into one single schema.

Five conformance classes are derived from this integrated model. Each conformance class is formally expressed by means of an EXPRESS model which uses whole or parts of the EXPRESS resource constructs belonging to the integrated information model defined in this part of ISO 13584. These conformance classes permit respectively the exchange of dictionaries with aggregates (compliant with the future release of IEC 61360-5), extended dictionaries with aggregates, extended dictionaries with library explicit content description and finally extended dictionaries with library explicit content description and aggregates. In the library description conformance classes, content of classes are described explicitly as sets of instances, without any constraint, variable or expression. This makes the description much simpler than in ISO 13584-24.

The integrated EXPRESS information models defined in this part of ISO 13584 contains provisions that permit a supplier library to reference external files. This part of ISO 13584 specifies which formats are allowed for these external files. Other external file formats are defined in the view exchange protocol series of parts of ISO 13584.

KEYWORDS: parts library, supplier library, digital library, electronic catalogue, exchange format, EXPRESS, information model, meta-specification, aggregate data types and values, explicit content description for libraries.

COMMENTS TO READER:

This document has been reviewed and noted by the ISO TC 184/SC4 Secretariat and has been determined to be ready for this ballot cycle.

The data models documented in this document are intended to be consistent with:

- ISO 13584-20:1998;
- ISO 13584-26:2000;
- ISO 13584-24 to be published
- ISO 13584-42:1998;
- IEC 61360-4 to be published

This set of documents constitutes the fifth release of P-LIB, suitable for implementation.

Project Leader: Gerald M. Radack

Address:

Concurrent Technologies Corporation
100 CTC Drive
Johnstown, PA 15904

USA

Telephone: +1 814 269 2679

Telefacsimile: +1 814 269 2402

Electronic mail: radack@ctc.com

Project Editor: Yamine Ait-Ameur

Address: ENSAE-SUPAERO

10 Avenue Edouard BELIN, BP 4032
31055 TOULOUSE CEDEX

France

Telephones: +33 5 62 17 80 57

Telefacsimile: +33 5 62 17 83 45

Electronic mail: yamine@supaero.fr

Contents

	Page
1. Scope.....	1
2. Normative references	1
3. Terms, definitions and abbreviations	2
4. Structure of ISO 13584-25.....	7
4.1. Generic resources	7
4.2. Library integrated model	7
5. Fundamental concepts and assumptions	9
5.1. Aggregate-structured value of properties	9
5.2. Explicit description of a class extension	9
6. ISO13584_IEC61360_dictionary_aggregate_extension_schema	10
6.1. Introduction to the ISO13584_IEC61360_dictionary_aggregate_extension_schema	10
6.3. ISO13584_IEC61360_dictionary_aggregate_extension_schema entity definitions	10
6.3.2. Aggregate_type.....	11
6.3.3. list_type	11
6.3.4. Set_type	12
6.3.5. Bag_type	13
6.3.6. Array_type.....	13
7. ISO13584_dictionary_aggregate_value_schema	14
7.1. Introduction to the ISO13584_IEC61360_dictionary_aggregate_value_schema	15
7.3. ISO13584_IEC61360_dictionary_aggregate_value_schema entity definitions	15
7.3.1. Aggregate_entity_instance_value	15
7.3.2. Aggregate_value.....	15
7.3.3. List_value	16
7.3.4. Set_value	16
7.3.5. Bag_value	16
7.3.6. Array_value	17
7.4. ISO13584_IEC61360_dictionary_aggregate_extension_schema function definitions	17
7.4.1. Compatible_aggregate_type_and_value function.....	17
7.4.2. Compatible_aggregate_domain_and_value function	18
7.4.3. Compatible_complete_types_and_value function	24
7.5. ISO13584_IEC61360_dictionary_aggregate_extension_schema rule definition.....	25
7.5.1. Allowed_aggregate_values rule	25
8. Library integrated information model 25	26
8.1. Conformance class requirements	27
8.1.1. Conformance class 0 : ISO13584_short_form_CC0_schema short listing	27
8.1.2. Conformance class 1 : ISO13584_short_form_CC1_schema short listing	31
8.1.3. Conformance class 2 : ISO13584_short_form_CC2_schema short listing	35
8.1.4. Conformance class 3 : ISO13584_short_form_CC3_schema short listing	40
8.1.5. Conformance class 4 : ISO13584_25_liim_schema short listing	46
Annex A (informative) ISO13584_25_liim_schema and conformance classes expanded listing.....	52
Annex B (normative) Standard data requirements for the library integrated information model 25	53
B.1 Constraints on a library delivery file for referencing library integrated information model 25	53
B.2 Conformance class specification table	54
B.3 Standard data for conformance class 0.....	54
B.4 Standard data for conformance class 1 to 4 (all the conformance classes but conformance class 0)	55
B.4.1. Constraints on a library delivery file conform to the library integrated model LIIM 25.....	55

Annex C (normative) Implementation method specific requirements for the library integrated information model 25	61
Annex D (informative) EXPRESS-G diagrams	62
Annex E (informative) Commented example of library integrated information model 25 physical files.	
Exchange of explicit general models	65
E.1 Capturing a parts family in ISO 13584	65
E.2 Description of the PAW parts family	67
E.2.1 Dictionary description: the BSU mechanism	67
E.2.2 Dictionary description: the dictionary element definition	67
E.2.3 Library specification: description of the class extension	68
E.3 A complete physical file for explicit general models.	69
Annex F (informative) Commented example of library integrated information model 25 physical files.	
Exchange of explicit functional models compliant with ISO 13584- 101	72
F.1 Description of the PAW parts family and its geometry	72
F.2. Description of geometric representations for the PAW parts family	73
F.3. Library specification of the functional model class.....	74
F.4 A complete physical file for explicit functional models compliant with ISO 13584-101.....	76
Annex G (informative) Commented example of library integrated information model 25 physical files.	
Exchange of explicit functional models compliant with ISO 13584- 102	89
G.1. Description of the PAW parts family and its geometry.....	89
G.2 A complete physical file for explicit functional models compliant with ISO 13584- 102	91
INDEX.....	105

Figures

Figure D. 1 — ISO13584_IEC61630_dictionary_aggregate_extension_schema diagram 1 of 1	63
Figure D. 2 — ISO13584_dictionary_aggregate_value_schema diagram 1 of 1	64
Figure E. 1 — PAW family description	65
Figure E. 2 — Instance of a dictionary description	66
Figure E. 3 — Explicit description of a dictionary description	66
Figure E. 4 — Identifiers of the concepts involved in the PAW family.....	67
Figure E. 5 — The BSU / Dictionary element relationship.....	67
Figure E. 6 — Dictionary_element of the concepts involved in the PAW family.....	68
Figure E. 7 — Dictionary_element of the concepts involved in the PAW family.....	69
Figure E. 7 — Dictionary_element of the concepts involved in the PAW family.....	69
Figure F.1 — The Identifiers of the concepts involved in the Paw family and its geometry representation	73
Figure F.2 — View control variables range definition	74
Figure F.3. — Specification of the view created by a functional model class.....	74
Figure F.4. — Description by extension of the instances of a functional a functional model	75

Figure F.5. — References to FORTRAN programs that display geometry.....	76
Figure F.6. — The BSU / Dictionary element relationship.....	76
Figure G. 14 — The functional model class extension with reference to STEP representations.....	91

Tables

Table 1 — Conformance options of library integrated information model 25	9
Table 2 — Conformance options of library integrated information model 25	27
Table B. 1 — ISO 13584 LIIM 25 conformance class specification	54

Foreword

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

Draft International Standards adopted by technical committees are circulated to the member bodies for voting. Publication as an International Standard requires approval by at least 75% of the member bodies casting a vote.

International Standard ISO 13584-24 was prepared by Technical Committee ISO/TC 184, Industrial automation system and integration, Subcommittee SC4, Industrial data.

ISO 13584 consists of the following parts under the general title *Industrial automation systems and integration - Parts library*:

- Part 1, Overview and fundamental principles;
- Part 10, Conceptual description: Conceptual model of parts library;
- Part 20, Logical resource: Logical model of expressions;
- Part 24, Logical resource: Logical model of supplier library;
- Part 25, Logical resource: Logical model of supplier library with aggregates-structured values and/or explicit content;
- Part 26, Logical resource: Information supplier identification;
- Part 31, Implementation resource: Geometric programming interface;
- Part 42, Description methodology: Methodology for structuring part families;
- Part 101, View exchange protocol: Geometric view exchange protocol by parametric program;
- Part 102, View exchange protocol: View exchange protocol by ISO 10303 conforming specification.

The structure of this International Standard is described in ISO 13584-1. The numbering of the parts of this International Standard reflects its structure:

- Parts 10 to 19 specify the conceptual descriptions,
- Parts 20 to 29 specify the logical resources,
- Parts 30 to 39 specify the implementation resources,
- Parts 40 to 49 specify the description methodology,
- Parts 50 to 59 specify the conformance testing,
- Parts 100 to 199 specify the view exchange protocol,

Should further parts of ISO 13584 be published, they will follow the same numbering pattern.

- Annex A is for information only.

- Annexes B and C form an integral part of this part of ISO 13584.

Introduction

ISO 13584 is an International Standard for the computer-interpretable representation and exchange of part library data. The objective is to provide a neutral mechanism capable of transferring parts library data, independent of any application that is using a parts library data system. The nature of this description makes it suitable not only for the exchange of files containing parts, but also as a basis for implementing and sharing databases of parts library data.

This International Standard is organized as a series of parts, each published separately. The parts of ISO 13584 fall into one of the following series: conceptual descriptions, logical resources, implementation resources, description methodology, conformance testing and view exchange protocol. The series are described in ISO 13584-1. This part of ISO 13584 is a member of the logical resources series.

This part of ISO 13584 specifies the generic resources needed for modeling supplier libraries that contain properties whose values may be aggregate-structured, and whose possible content is explicitly described as a set of instances. It also provides the EXPRESS integrated information models that permit the exchange of such supplier libraries. Knowledge of EXPRESS as defined in ISO 10303-11:1994 is required to understand this part of ISO 13584. Basic knowledge of ISO 13584-10: ISO 13584-24:—, and ISO 13584-42:1998 is also required.

The generic resources specified in this document were developed as a joint effort of ISO TC184/SC4/WG2 and IEC SC3D. They are intended to be documented both in this part of ISO 13584 and in IEC 61360-5. Both committees agreed not to change and/or modify the presented EXPRESS schemas independent of each other in order to guarantee the harmonization and the reusability of the work from both committees. Requests for amendments should therefore be sent to both committees. These requests should be adopted by both committees before modifying the EXPRESS schemas.

Industrial automation systems and integration – Parts library – Part 25: Logical resources: Logical model of supplier library with aggregate values and explicit content

1. Scope

This part of ISO 13584 provides generic EXPRESS resource constructs that support the description of aggregate data types and values occurring in supplier libraries. It also contains an integrated EXPRESS information model for representing supplier libraries for the purpose of exchange. This integrated information model integrates the above resource constructs with other EXPRESS resource constructs from different parts of ISO 13584 and ISO 10303 into one single schema. Supplier libraries may consist of definitions and of representations of families of parts. They may also define new representation categories. Supplier libraries may consist only of dictionary elements with or without aggregate data types, or they may also contain explicit specifications of the sets of permitted instances.

When used together with view exchange protocols, this integrated information model also permits the exchange of one or several representation categories for the parts defined in a parts library.

The following are within the scope of this part of ISO 13584:

- generic resource constructs for representing aggregate data types. Aggregate data types and values are modeled according to the definition of aggregate data types of the EXPRESS language (ISO 10303-11: 1994);
- generic resource constructs for representing aggregate values;
- description of assembled parts that may contain an unlimited number of constituent components;
- a library integrated information model that provides for modeling and exchanging supplier libraries that contain properties whose values may be aggregate-structured, and whose possible class extensions are explicitly described as sets of instances.

The following are outside the scope of this part of ISO 13584:

- representation of expressions and variables;
- implicit description of the set of permitted instances of a class by means of constraints;
- specification of a software system able to manage supplier libraries represented according to the information models defined in this part of ISO 13584.

2. Normative references

The following normative documents contain provisions which, through reference in this text, constitute provisions of this part of ISO 13584. For dated references, subsequent amendments to, or revisions of, any of these publications do not apply. However, parties to agreements based on this part of ISO

13584 are encouraged to investigate the possibility of applying the most recent editions of the normative documents indicated below. For undated references the latest edition of the publication referred to applies. Members of ISO and IEC maintain registers of currently valid International Standards.

IEC 61360-2: 1998, *Standard data element types with associated classification scheme for electric components - Part 2: EXPRESS dictionary schema*

ISO 10303-11: 1994, *Industrial automation systems and integration - Product data representation and exchange - Part 11: Description methods: The EXPRESS language reference manual*.

ISO 10303-21: 1994, *Industrial automation systems and integration - Product data representation and exchange - Part 21: Implementation methods: Clear text encoding of the exchange structure*.

ISO 10303-41: 1994, *Industrial automation systems and integration - Product data representation and exchange - Part 41: Integrated generic resources: Fundamentals of product description and support*.

ISO 10303-42: 1994, *Industrial automation systems and integration - Product data representation and exchange - Part 42: Integrated generic resources: Geometric and topological representation*.

ISO 13584-24:^{—1}, *Industrial automation systems and integration - Parts library - Part 24: Logical resource: Logical model of supplier library*.

ISO 13584-42:1998, *Industrial automation systems and integration - Parts library - Part 42: Description methodology: Methodology for structuring part families*

3. Terms, definitions and abbreviations

For the purposes of this part of ISO 13584, the following terms and definitions apply. Some of these terms and definitions are repeated for convenience from:

- ISO 10303-1:1994;
- ISO 10303-11:1994;
- ISO 13584-1:^{—1};
- ISO 13584-10:^{—1};
- ISO 13584-42: 1998.
- ISO 13584-24:^{—1}.

3.1.

applicable property

a property that is defined for some family of parts and that shall apply to any part that belongs to this family of parts.

[ISO 13584-10: ^{—1}]

EXAMPLE For a screw generic family of parts, the threaded diameter is an applicable property: this characteristic applies to any screw.

¹ To be published

3.2.**basic semantic unit****BSU**

the entity that provides an absolute and universal identification of certain objects of the application domain (e.g. classes, data element types).

[ISO 13584-42:1998, definition 3.4.1]

3.3.**class extension**

set of all the different possible instances conforming to the specification defined by a class.

[ISO 13584-24: —¹]

3.4.**common dictionary schema**

the information model for a dictionary, using the EXPRESS modeling language, resulting from a joint effort between ISO TC184/SC4/WG2 and IEC SC3D.

[ISO 13584-42:1998, definition 3.4.3]

NOTE The common dictionary schema is specified in IEC 61360-2, and its content is provided in the informative annex D of ISO 13584-42:1998.

3.6.**conformance class**

a subset of a standard for which conformance may be claimed.

[ISO 13584-24: —¹]

3.8.**conformance requirement**

a precise, text definition of a characteristic required to be present in a conforming implementation.

[ISO 10303-1:1994, definition 2.1.14]

3.10.**dictionary element**

the set of attributes that constitutes the dictionary description of certain objects of the application domain (e.g. classes, data element types).

[ISO 13584-42:1998, definition 3.4.7]

3.12.**data element type****DET**

unit of data for which the identification, the description and value representation have been specified.

[ISO 13584-42:1998, definition 3.4.4]

3.13.**data type**

a domain of values.

[ISO 10303-11:1994, definition 3.2.4]

¹ To be published

3.15.

family of parts

a simple or generic family of parts.
[ISO 13584-10: —¹]

3.16.

functional model of a part

the library data that represent one representation category of a part in an integrated library.
[ISO 13584-1: —¹]

3.17.

functional view of a part

the data that represent one representation category of a part in product data.
[ISO 13584-1: —¹]

NOTE The structure of a functional view does not depend on the part it represents.

3.18.

general model of a part

the library data that carries the definition and identity of a part in an integrated library.
[ISO 13584-1: —¹]

3.19.

generic family of parts

a grouping of simple or generic families of parts done for purposes of classification or for factoring common information.
[ISO 13584-10: —¹]

3.20.

library delivery file

a population of EXPRESS entity instances conforming to a library integrated information model and represented according to one of the implementation methods specified in ISO 10303.

NOTE A library delivery file specifies the structure and the content of a supplier library. It may reference library external files.

3.21.

library part

a part associated with a set of data that represents it in a library.
[ISO 13584-1: —¹]

3.22.

library part data

the data that represent a part in a library.

¹ To be published

[ISO 13584-1: —¹]

3.23.

library exchange context

the set of one library delivery file and zero, one or several library external files that represent together a supplier library.

3.24.

library external file

a file, referenced from a library delivery file, that contributes to the definition of a supplier library.

NOTE The structure and the format of a library external file is specified in the library delivery file that references it.

3.25.

library integrated information model

LIM

an EXPRESS schema that integrates resource constructs from different EXPRESS schemas for representing supplier libraries for the purpose of exchange and that is associated with conformance requirements.

3.26.

library specification of a class

explicit representation of a class extension in a supplier library

NOTE 1 In the ISO 13584 series, every class is intentionally defined through a dictionary element. Only those classes of which the supplier desires to represent explicitly the possible instances are associated with a library specification.

NOTE 2 In this part of ISO 13584, the library specification of a class consists of a set that contains all the different possible instances.

3.27.

part

material or functional element that is intended to constitute a component of different products.

[ISO 13584-1: —¹]

3.28.

property

an information that may be represented by a data element type.

[ISO 13584-42:1998, definition 3.4.10]

3.29.

representation category

an abstraction used to distinguish between different possible user requirements regarding a part representation.

[ISO 13584-1: —²]

¹ To be published

² To be published

NOTE In the model defined in the ISO 13584 standard series, this distinction is formally expressed in terms of a view logical name and in terms of the view control variables.

3.30.

resource construct

the collection of EXPRESS language entities, types, functions, rules and references that together define a valid description of data.

[ISO 13584-24: —¹]

3.31.

simple family of parts

a set of parts of which each part may be described by the same group of properties.

[ISO 13584-10: —¹]

3.32.

supplier library

a set of data, and possibly of programs, for which the supplier is defined and that describes in the standard format defined in ISO 13584 a set of parts and/or a set of representation of parts.

[ISO 13584-1: —¹]

3.33.

view exchange protocol

VEP

a part of ISO 13584 that describes the use of resource constructs and of representation transmission interfaces that satisfy the information requirement for the exchange of one representation category of parts.

3.34.

visible property

a property that is defined for some family of parts and that may or not apply to the different parts of this family of parts.

[ISO 13584-10: —¹]

EXAMPLE For a generic family of screws, the non-threaded length is a visible property: it is clearly defined for any screw, but only those screws with a non-threaded part have a value for this property.

NOTE 1 The code of the class where a property is defined as visible is part of the identification of the data element type that represents this property.

4. Structure of ISO 13584-25

ISO 13584-25 has two main parts.

- The generic resources part provides resource constructs for representing aggregate data types and values. Aggregate data types and values are modeled in total conformance with the EXPRESS language.
- The library integrated information model gathers the above resource constructs with other generic resource constructs from the LIIM 25 different parts of ISO 13584 and ISO 10303 into one single schema for representing supplier libraries that may include aggregate data types and aggregate values and where possible class extensions are explicitly described as sets of instances.

4.1. Generic resources

The generic resources consist of the following EXPRESS schemas:

- **ISO13584_IEC61360_dictionary_aggregate_extension_schema;**
- **ISO13584_dictionary_aggregate_value_schema;**

These schemas provide resource constructs that are generic in nature. They may be used outside ISO13584, and particularly in all the applications that use a data dictionary compliant with the IEC 61360 standard series.

4.1.1. ISO13584_IEC61360_dictionary_aggregate_extension_schema

The **ISO13584_IEC61360_dictionary_aggregate_extension_schema** provides the resource constructs needed to describe data types corresponding to aggregate data types as defined in the EXPRESS language. It defines resources to describe array, bag, list and set data types. These data types extend the data types already defined in the **ISO13584_ISO61360_dictionary_schema** published in IEC 61360-2 and whose content is duplicated in an informative annex of ISO 13584-42:1998.

4.1.2. ISO13584_dictionary_aggregate_value_schema

The **ISO13584_dictionary_aggregate_value_schema** provides the resource constructs needed to describe values of data types corresponding to aggregate data types as defined in the EXPRESS language. It defines resources to describe array, bag, list and set-structured values. These data values extend the data values already defined in the **ISO13584_instance_resource_schema** specified in ISO 13584-24.

4.2. Library integrated model

The library integrated information model specified in this part of ISO 13584 allows to use the resource constructs from the above schemas for representing supplier libraries for the purpose of exchange. The library integrated information model defined in this part of ISO 13584 enables the exchange of five kinds of libraries between a library data supplier and a library end-user:

- dictionaries that define hierarchies of classes of parts, materials or other items, with aggregate-structured properties,
- dictionaries that define hierarchies of classes of items, of item representations, and of representation categories of items, without aggregate-structured properties,
- dictionaries that define hierarchies of classes of items, of item representations, and of representation categories of items, with aggregate-structured properties,
- libraries that specify the set of instances that belong to hierarchies of classes of items and of item representations, without aggregate-structured properties, and
- libraries that specify the set of instances that belong to hierarchies of classes of items and of item representations, with aggregate-structured properties,

Each of these kind of libraries corresponds to one conformance class of this part of ISO 13584. Each conformance class specifies the conformance requirements for implementations that claim conformance to this conformance class. Conformance class 5 specifies all the entities, types and associated constructs that are part of the library integrated information model LIIM 25. Other conformance classes shall only support a subset of this set of resource constructs. In this part of ISO 13584, each subset that defines a conformance class is formally defined by means of an

¹ To be published

EXPRESS schema. The following schemas specify the five conformance classes defined in this part of ISO 13584.

4.2.1 ISO13584_short_form_CC0_schema

The **ISO13584_short_form_CC0_schema** specifies the information requirements for exchanging definitions of hierarchies of item classes, where items may be parts, materials or features. This integrated model allows the exchange of all the dictionary elements from the ISO/IEC dictionary more item classes whose properties may have aggregate-structured values. Conformance requirements to conformance class 0 of LIIM 25 are defined in Clause 8.1.1 of this part of ISO 13584.

4.2.2 ISO13584_short_form_CC1_schema

The **ISO13584_short_form_CC1_schema** specifies the information requirements for exchanging definitions of hierarchies of item classes, where items may be parts, materials or features together with definitions of representations of such item classes. This schema is associated with a set of standard data, that defines the formats of library external files that may be referenced by a library delivery file conform to conformance class 1 of LIIM 25, and with implementation methods for the library delivery file. Together with the standard data specified in annex B and with the implementation methods specified in annex C, **ISO13584_short_form_CC1_schema** specifies conformance class 1 of the library integrated information model LIIM 25. Conformance requirements to conformance class 1 of LIIM 25 are defined in Clause 8.1.2 of this part of ISO 13584.

4.2.3 ISO13584_short_form_CC2_schema

The **ISO13584_short_form_CC2_schema** specifies the information requirements for exchanging definitions of hierarchies of item classes, where items may be parts, materials or and definitions of functional models and functional views whose properties may have aggregate-structured values. This schema is associated with a set of standard data, that defines the formats of library external files that may be referenced by a library delivery file conform to conformance class 2 of LIIM 25, and with implementation methods for the library delivery file. Together with the standard data specified in annex B and with the implementation methods specified in annex C, **ISO13584_short_form_CC2_schema** specifies conformance class 2 of the library integrated information model LIIM 25. Conformance requirements to conformance class 2 of LIIM 25 are defined in Clause 8.1.3 of this part of ISO 13584.

4.2.4 ISO13584_short_form_CC3_schema

The **ISO13584_short_form_CC3_schema** specifies the information requirements for exchanging definitions and instances of hierarchies of item classes, where items may be parts, materials or features together with definitions and instances of functional models and functional views that represent instances of such item classes. Class extensions may only be defined by sets of instances, without any constraint or expression. This schema is associated with a set of standard data, that defines the formats of library external files that may be referenced by a library delivery file conform to conformance class 3 of LIIM 25, and with implementation methods for the library delivery file. Together with the standard data specified in annex B and with the implementation methods specified in annex C, **ISO13584_short_form_CC3_schema** specifies conformance class 3 of the library integrated information model LIIM 25. Conformance requirements to conformance class 3 of LIIM 25 are defined in Clause 8.1.4 of this part of ISO 13584.

4.2.5 ISO13584_short_form_CC4_schema

The **ISO13584_short_form_CC4_schema** specifies the information requirements for exchanging hierarchies of item classes, where items may be parts, materials or features with definitions and instances of functional models and functional views, with aggregate types and values and explicit class extensions. Class extensions may only be defined by sets of instances, without any constraint or expression. This schema is associated with a set of standard data, that defines the formats of library external files that may be referenced a library delivery file conform to conformance class 4 of LIIM 25, and with implementation methods for the library delivery file. Together with the standard data specified in annex B and with the implementation methods specified in annex C,

ISO13584_short_form_CC1_schema specifies conformance class 4 of the library integrated information model LIIM 25. Conformance requirements to conformance class 4 of LIIM 25 are defined in Clause 8.1.5 of this part of ISO 13584.

Table 1 — Conformance options of library integrated information model 25

Capabilities	Dictionary_elements			Library specification (class extension)
Conformance class	Dictionary definitions of item classes	Dictionary definitions of item classes and representation	Aggregate structured properties	
0	x		x	
1	x	x		
2	x	x	x	
3	x	x	x	x
4	x	x	x	x

5. Fundamental concepts and assumptions

The following concepts and assumptions apply to this part of ISO 13584.

5.1. Aggregate-structured value of properties

The **ISO13584_IEC61360_dictionary_aggregate_extension_schema** and **ISO13584_dictionary_aggregate_value_schema** schemas define the generic resource constructs for representing aggregate data types and generic resource constructs for representing aggregate values. Aggregate data types and values are modeled according to the definition of aggregate data types of the EXPRESS language (ISO 10303-11: 1994). These resources allows to assign aggregate type and values to any defined resource that shall be typed or valued by an aggregate.

5.2. Explicit description of a class extension

The library integrated information model defined in this part of ISO 13584 provides for modeling and exchanging supplier libraries that contain properties whose values may be aggregate-structured, and whose possible class extensions are explicitly described as sets of instances. This library integrated information model is provided for the explicit representation of libraries by a set of instances (**lib_item_instances**) associated to properties. The resources allowing to describe such libraries are provided in ISO 13584 Part 24.

6. ISO13584_IEC61360_dictionary_aggregate_extension_schema

This clause defines the requirements for the **ISO13584_IEC61360_dictionary_aggregate_extension_schema**. The following EXPRESS declaration introduces **ISO13584_IEC61360_dictionary_aggregate_extension_schema** block and identifies the necessary external references.

EXPRESS specification:

*)

```
SCHEMA ISO13584_IEC61360_dictionary_aggregate_extension_schema;
```

```
REFERENCE FROM ISO13584_IEC61360_dictionary_schema(
    data_type,
    entity_instance_type ) ;
(*
```

NOTE The schema referenced above can be found in the following document:
ISO13584_IEC61360_dictionary_schema IEC 61360-2:1998
 (which is duplicated for convenience in Informative Annex D of ISO 13584-42: 1998)

6.1. Introduction to the ISO13584_IEC61360_dictionary_aggregate_extension_schema

This provides the information model for the extension to the ISO/IEC common dictionary schema which extends the scope of **entity_instance_type** data types to allow the use of lists, sets, bags, arrays of simple or complex data types.

This extension is achieved in two steps.

- the **entity_instance_type_for_aggregate** provides for referencing a EXPRESS-defined entities that specify aggregate data types. The **entity_instance_type_for_aggregate** entity is obtained by subtyping the **entity_instance_type**.

NOTE The **entity_instance_type** entity is defined in the IEC 61630 –2:1998 and duplicated in ISO 13584 –42:1998.

- then, entities that specify aggregate data types are modeled by the **aggregate_type** entity and its specialisations.

6.3. ISO13584_IEC61360_dictionary_aggregate_extension_schema entity definitions

The following entity type definitions describe the necessary resources needed to encode aggregate types according to the EXPRESS language.

6.3.1. Aggregate_entity_instance_type

The **entity_instance_type_for_aggregate** entity provides for referencing definitions of data types which may be expressed as lists, sets, bags or arrays of simple or complex values. It is defined by referencing an **aggregate_type** defined in this schema.

EXPRESS specification:

```
*)
ENTITY entity_instance_type_for_aggregate

SUBTYPE OF ( entity_instance_type ) ;
    type_structure : aggregate_type;
WHERE
    WR1: SELF\entity_instance_type.type_name=
        ['ISO13584_IEC61360_DICTIONARY_EXTENSION_SCHEMA' +
         'AGGREGATE_TYPE'] ;
END_ENTITY;
(*)
```

Attribute definition:

type_structure : the **aggregate_type** referenced and carried by the **entity_instance_type**.

Formal propositions:

WR1: the **type_name** attribute of the **entity_instance_type** shall contain the string 'ISO13584_IEC61360_DICTIONARY_EXTENSION_SCHEMA.AGGREGATE_TYPE'.

6.3.2. Aggregate_type

The **aggregate_type** entity provides for the definition of data types which may be expressed as lists, sets, bags or arrays of simple or complex values.

EXPRESS specification:

```
*)
ENTITY aggregate_type

ABSTRACT SUPERTYPE OF ( ONEOF (
    list_type,
    set_type,
    bag_type,
    array_type) );
    value_type : data_type;
END_ENTITY;
(*
```

Attribute definition:

value_type : is the type of value (simple or complex) which is used for each element of the aggregate

6.3.3. list_type

The **list_type** entity provides for the definition of data types which may be expressed as ordered lists of values in which duplication may or may not be allowed..

EXPRESS specification:

```
*)
ENTITY list_type

SUBTYPE OF (aggregate_type);
    start_index, end_index : OPTIONAL integer;
    uniqueness : Boolean;
WHERE
    WR1 : start_index <= end_index;
    WR2 : (EXISTS(start_index) AND NOT(EXISTS(end_index)))
        OR (EXISTS(start_index) AND (EXISTS(end_index)))
        OR (NOT (EXISTS(start_index)) AND NOT(EXISTS(end_index)));
END_ENTITY;
(*)
```

Attribute definition:

start_index : the index number of the first element of the list.

end_index : the index number of the last element of the list.

uniqueness : a flag to indicate whether all elements of the list must be unique (true) or whether duplicates are allowed (false).

Formal propositions:

WR1: **end_index** cannot be less than **start_index**.

WR2: If **end_index** is available, then **start_index** is available as well.

6.3.4. Set_type

The **set_type** entity provides for the definition of data types which may be expressed as unordered collections of values in which no duplication can occur.

EXPRESS specification:

```
*)  
ENTITY set_type  
  
SUBTYPE OF (aggregate_type);  
    start_index, end_index : OPTIONAL integer;  
WHERE  
    WR1 : start_index <= end_index;  
    WR2 : (EXISTS(start_index) AND NOT(EXISTS(end_index)))  
          OR (EXISTS(start_index) AND (EXISTS(end_index)))  
          OR (NOT (EXISTS(start_index)) AND NOT(EXISTS(end_index)));  
END_ENTITY;  
(*
```

Attribute definition:

start_index : the index number of the first element of the list.

end_index : the index number of the last element of the list.

Formal propositions:

WR1: **end_index** cannot be less than **start_index**.

WR2: If **end_index** is available, then **start_index** is available as well.

6.3.5. Bag_type

The **bag_type** entity provides for the definition of data types which may be expressed as unordered collections of values in which duplication may occur.

EXPRESS specification:

```
*)  
ENTITY bag_type
```

```

SUBTYPE OF (aggregate_type);
    start_index, end_index : OPTIONAL integer;
WHERE
    WR1 : start_index <= end_index;
    WR2 : (EXISTS(start_index) AND NOT(EXISTS(end_index)))
        OR (EXISTS(start_index) AND (EXISTS(end_index)))
        OR (NOT (EXISTS(start_index)) AND NOT(EXISTS(end_index)));
END_ENTITY;
(*)

```

Attribute definition:

start_index : the index number of the first element of the list

end_index : the index number of the last element of the list

Formal propositions:

WR1: **end_index** cannot be less than **start_index**.

WR2: if **end_index** is available, then **start_index** is available as well.

6.3.6. Array_type

The **array_type** entity provides for the definition of data types which may be expressed as an array of values which may or may not be unique and in which individual elements may be identified by an integer index.

EXPRESS specification:

```

*)
ENTITY array_type

SUBTYPE OF (aggregate_type);
    start_index, end_index : integer;
    uniqueness : BOOLEAN;
WHERE
    WR1 : start_index <= end_index;
    WR2 : (EXISTS(start_index) AND EXISTS(end_index));
END_ENTITY;
(*)

```

Attribute definition:

start_index : the index number of the first element of the list.

end_index : the index number of the last element of the list.

uniqueness : a flag to indicate whether all elements of the list must be unique (true) or whether duplicates are allowed (false).

Formal propositions:

WR1: **end_index** cannot be less than **start_index**.

WR2: If **end_index** is available, then **start_index** is available as well.

*)

```
END_SCHEMA ; -- ISO13584_IEC61360_dictionary_aggregate_extension_schema
(*)
```

7. ISO13584_dictionary_aggregate_value_schema

This clause defines the requirements for the **ISO13584_IEC61360_dictionary_aggregate_value_schema**. The following EXPRESS declaration introduces **ISO13584_IEC61360_dictionary_aggregate_value_schema** block and identifies the necessary external references.

EXPRESS specification:

```
*)  
SCHEMA ISO13584_dictionary_aggregate_value_schema;  
  
REFERENCE FROM ISO13584_IEC61360_dictionary_schema(data_type,  
                                                 data_type_element,  
                                                 level_type,  
                                                 property_BSU,  
                                                 property_DET);  
  
REFERENCE FROM ISO13584_IEC61360_dictionary_aggregate_extension_schema;  
  
REFERENCE FROM ISO13584_extended_dictionary_schema (data_type_typeof);  
  
REFERENCE FROM ISO13584_instance_resource_schema (  
                                                 compatible_class_and_class,  
                                                 compatible_level_type_and_instance,  
                                                 compatible_type_and_value,  
                                                 primitive_value,  
                                                 property_or_data_type_BSU,  
                                                 property_value,  
                                                 uncontrolled_entity_instance_value) ;  
(*
```

NOTE The schemas referenced above can be found in the following documents:
ISO13584_IEC61360_dictionary_schema IEC 61360-2:1998
 (which is duplicated for convenience in Informative Annex D of ISO 13584-42: 1998)
ISO13584_IEC61360_dictionary_aggregate_extension_schema This part of ISO,
ISO13584_instance_resource_schema ISO 13584-24: 2001,

7.1. Introduction to the ISO13584_IEC61360_dictionary_aggregate_value_schema

The **ISO13584_dictionary_aggregate_value_schema** provides the information model for the extension to the **ISO_13584_instance_resource_schema** which extends the scope of instance values to instances whose data types are aggregate data types that are lists, sets, bags or arrays of simple or complex data types, as allowed by the extension of the ISO/IEC common dictionary schema defined in the **ISO13584_IEC61360_dictionary_aggregate_extension_schema** schema.

7.3. ISO13584_IEC61360_dictionary_aggregate_value_schema entity definitions

The following entity type definitions describe the necessary resources needed to encode aggregate values according to the EXPRESS language.

7.3.1. Aggregate_entity_instance_value

The **aggregate_entity_instance_value** entity provides for the referencing of data values which may be expressed as aggregates of **primitive_values**.

It is obtained by subtyping the **uncontrolled_entity_instance_value** provided by the **ISO13584_instance_resource_schema** defined in ISO 13584-24: 2001.

EXPRESS specification:

```
*)
ENTITY aggregate_entity_instance_value

SUBTYPE OF (uncontrolled_entity_instance_value) ;
    the_value : aggregate_value ;
END_ENTITY;
(*
```

Attribute definition:

the_value : the aggregate carrying all the values of the described entity.

7.3.2. Aggregate_value

The **aggregate_value** entity provides for the definition of data values which may be expressed as aggregates of **primitive_values**.

EXPRESS specification:

```
*)
ENTITY aggregate_value

ABSTRACT SUPERTYPE OF ( ONEOF (
    list_value,
    set_value,
    bag_value,
    array_value)) ;
    values : aggregate OF primitive_value ;
END_ENTITY;
(*)
```

Attribute definition:

values : is the aggregate carrying all the values of the described entity.

7.3.3. List_value

The **list_value** entity provides for the definition of data values which may be expressed as lists of **primitive_or_aggregate_values**.

EXPRESS specification:

```
*)
ENTITY list_value
```

```
SUBTYPE OF (aggregate_value) ;
  SELF\aggregate_value.values : LIST OF primitive_value ;
END_ENTITY;
(*)
```

Attribute definition:

values : the list carrying all the values of the described entity.

7.3.4. Set_value

The **set_value** entity provides for the definition of data values which may be expressed as sets of **primitive_or_aggregate_values**.

EXPRESS specification:

```
*)
ENTITY set_value

SUBTYPE OF (aggregate_value) ;
  SELF\aggregate_value.values : SET OF primitive_value ;
END_ENTITY;
(*)
```

Attribute definition:

values : the set carrying all the values of the described entity.

7.3.5. Bag_value

The **bag_value** entity provides for the definition of data values which may be expressed as bags of **primitive_or_aggregate_values**.

EXPRESS specification:

```
*)
ENTITY bag_value

SUBTYPE OF (aggregate_value) ;
  SELF\aggregate_value.values : BAG OF primitive_value ;
END_ENTITY;
(*)
```

Attribute definition:

values : is the bag carrying all the values of the described entity.

7.3.6. Array_value

The **array_value** entity provides for the definition of data values which may be expressed as arrays of **primitive_or_aggregate_values**.

EXPRESS specification:

```
*)
```

```

ENTITY array_value

SUBTYPE OF (aggregate_value) ;
    low_bound, high_bound : INTEGER ;
    SELF\aggregate_value.Values : ARRAY[low_bound : high_bound] OF
        primitive_value;
END_ENTITY;
(*

```

Attribute definition:

low_bound : the low bound value of the described array.

high_bound : the high bound value of the described array.

values : the array carrying all the values of the described entity.

7.4. ISO13584_IEC61360_dictionary_aggregate_extension_schema function definitions

7.4.1.Compatible_aggregate_type_and_value function

The **compatible_aggregate_type_and_value** function checks that a pair of a **property_or_data_type_BSU** and a **primitive_value** are type compatible. It returns UNKNOWN if the type of a **property_BSU** is not available.

This function retrieves the final **data_type** data type and calls the **compatible_aggregate_domain_and_value** function.

EXPRESS specification:

```

*)
FUNCTION compatible_aggregate_domain_and_value (
    dom : property_or_data_type_BSU ;
    val : primitive_value )
    : LOGICAL ;

LOCAL
    the_data_type : data_type ;
END_LOCAL ;
-- Checking of the availability of the final type of a property or a
-- data type BSU.
IF 'ISO13584_IEC61360_DICTIONARY_SCHEMA.PROPERTY_BSU' IN TYPEOF(dom)
THEN
    IF NOT(SIZEOF(dom.definition)=0) THEN
        The_data_type:= dom.definition[1]\property_DET.domain ;
    ELSE
        RETURN (UNKNOWN) ;
    END_IF ;
END_IF ;
IF 'ISO13584_IEC61360_DICTIONARY_SCHEMA.DATA_TYPE_BSU' IN TYPEOF(dom)
THEN
    IF NOT(SIZEOF(dom.definition)=0) THEN
        the_data_type :=dom.definition[1]\
            data_type_element.type_definition;

```

```

        ELSE
            RETURN (UNKNOWN) ;
        END_IF ;
    END_IF ;
    RETURN(compatible_aggregate_domain_and_value (the_data_type , val));

END_FUNCTION ;
(*

```

7.4.2.Compatible_aggregate_domain_and_value function

The **compatible_aggregate_domain_and_value** function checks that a **data_type** is the correct type for a **val primitive_value**. It returns UNKNOWN if the type of a **property_BSU** is not available.

This function checks that the length and the low and high bounds of an aggregate are compatible with the type definition.

Moreover, it checks that the values of the aggregate are compatible with their type definition.

EXPRESS specification:

```

*)
FUNCTION compatible_aggregate_domain_and_value (
    The_data_type: data_type;
    val : primitive_value )
    : LOGICAL ;

LOCAL
    type_of_elements      : data_type;
    elements              : aggregate of primitive_value ;
    at_least_one_is_uncontrolled, Aggregates_are_ok,
    are_aggregates: BOOLEAN ;
    Result : LOGICAL ;
END_LOCAL ;

-- Checking that the type is an entity_instance_type whose
-- type_name_attribute references the aggregate_type entity.

IF (NOT('ISO13584_IEC61360_DICTIONARY_SCHEMA.' +
        'ENTITY_INSTANCE_TYPE' IN TYPEOF(the_data_type)))
THEN
    RETURN(UNKNOWN) ;
END_IF ;

IF NOT(['ISO13584_IEC61360_DICTIONARY_EXTENSION_SCHEMA.' +
        'AGREGATE_TYPE'] = the_data_type.type_name)
THEN
    RETURN(UNKNOWN) ;
END_IF ;

-- type_of_elements that contain the type of the elements of the

```

```

-- aggregate_type data type.
type_of_elements :=the_data_type.type_structure.value_type ;

-- Checking that the values of val are
--aggregate_entity_instance_values.

IF NOT ('ISO13584_DICTIONARY_AGGREGATE_VALUE_SCHEMA.' +
        'AGGREGATE_ENTITY_INSTANCE_VALUE' IN TYPEOF (val))
THEN
    RETURN(UNKNOWN) ;
END_IF ;

-- elements contains the aggregate of the values.
elements :=
val\aggregate_entity_instance_value.the_value\aggregate_value.values ;

-- checking that that low and high bounds of the values are compatible
-- with the type declaration.

IF (
('ISO13584_DICTIONARY_AGGREGATE_VALUE_SCHEMA.' +
     'LIST_VALUE' IN TYPEOF (val)) OR
('ISO13584_DICTIONARY_AGGREGATE_VALUE_SCHEMA.' +
     'BAG_VALUE' IN TYPEOF (val)) OR
('ISO13584_DICTIONARY_AGGREGATE_VALUE_SCHEMA.' +
     'SET_VALUE' IN TYPEOF (val))
)
AND (
    (EXISTS(the_data_type.type_structure.start_index))
    AND NOT
    (SIZEOF(elements) >=the_data_type.type_structure.start_index)
)
THEN
    RETURN(FALSE) ;
END_IF ;

IF (
('ISO13584_DICTIONARY_AGGREGATE_VALUE_SCHEMA.' +
     'LIST_VALUE' IN TYPEOF (val)) OR
('ISO13584_DICTIONARY_AGGREGATE_VALUE_SCHEMA.' +
     'BAG_VALUE' IN TYPEOF (val)) OR
('ISO13584_DICTIONARY_AGGREGATE_VALUE_SCHEMA.' +
     'SET_VALUE' IN TYPEOF (val))
)
AND (
    (EXISTS(the_data_type.type_structure.end_index))
    AND NOT
    (HIINDEX(elements) <=the_data_type.type_structure.end_index)
)
THEN
    RETURN(FALSE) ;
END_IF ;

```

```

--For array data type and values, checking that low and high bounds
-- of the values are compatible with the type declaration.

IF
    ('ISO13584_DICTIONARY.Aggregate_Value_SCHEMA.' +
     'ARRAY_VALUE' IN TYPEOF (val))
AND
(
    NOT ('ISO13584_IEC61360_DICTIONARY_AGGREGATE_EXTENSION_SCHEMA.' +
          'ARRAY_TYPE' IN TYPEOF (the_data_type.type_structure))
    OR
    NOT (the_data_type.type_structure.start_index =
        val\aggregate_entity_instance_value.the_value\array_value.low_bound)
    OR
    NOT (the_data_type.type_structure.end_index =
        val\aggregate_entity_instance_value.the_value\array_value.high_bound )
)
THEN
    RETURN(FALSE) ;
END_IF ;

-- Checking of the compatibility of the values in the aggregate.
-- Aggregate of integers
IF 'ISO13584_IEC61360_DICTIONARY_SCHEMA.INTEGER_TYPE'
    IN TYPEOF(type_of_elements )
THEN
    REPEAT i :=LOINDEX(elements) TO HIINDEX(elements) ;
        IF NOT
            ('ISO13584_INSTANCE_RESOURCE_SCHEMA.INTEGER_VALUE'
             IN TYPEOF(elements[i]))
        THEN
            RETURN(FALSE) ;
        END_IF ;
    END_REPEAT ;
END_IF ;

-- Aggregate of Reals
IF ('ISO13584_IEC61360_DICTIONARY_SCHEMA.REAL_TYPE'
    IN TYPEOF(type_of_elements ))
THEN
    REPEAT i :=LOINDEX(elements) TO HIINDEX(elements) ;
        IF NOT
            ('ISO13584_INSTANCE_RESOURCE_SCHEMA.REAL_VALUE'
             IN TYPEOF(elements[i]))
        THEN
            RETURN(FALSE) ;
        END_IF ;
    END_REPEAT ;
END_IF ;

-- Aggregate of Booleans
IF ('ISO13584_IEC61360_DICTIONARY_SCHEMA.BOOLEAN_TYPE'

```

```

    IN TYPEOF(type_of_elements ))
THEN
    REPEAT i :=LOINDEX(elements) TO HIINDEX(elements) ;
        IF NOT
            ('ISO13584_INSTANCE_RESOURCE_SCHEMA.BOOLEAN_VALUE'
            IN TYPEOF(elements[i]))
        THEN
            RETURN(FALSE) ;
        END_IF ;
    END_REPEAT ;
END_IF ;

-- Aggregate of Strings
IF ('ISO13584_IEC61630_DICTIONARY_SCHEMA.STRING_TYPE'
    IN TYPEOF(type_of_elements ))
THEN
    REPEAT i :=LOINDEX(elements) TO HIINDEX(elements) ;
        IF NOT
            ('ISO13584_INSTANCE_RESOURCE_SCHEMA.STRING_VALUE'
            IN TYPEOF(elements[i]))
        THEN
            RETURN(FALSE) ;
        END_IF ;
    END_REPEAT ;
END_IF ;

-- Aggregate of entity instances
IF ('ISO13584_IEC61630_DICTIONARY_SCHEMA.ENTITY_INSTANCE_TYPE'
    IN TYPEOF(type_of_elements ))
THEN
    at_least_one_is_uncontrolled :=FALSE ;
    REPEAT i :=LOINDEX(elements) TO HIINDEX(elements) ;
        IF
            ('ISO13584_INSTANCE_RESOURCE_SCHEMA.'+
            'UNCONTROLLED_ENTITY_INSTANCE_VALUE')
            IN TYPEOF(elements[i])
        THEN
            at_least_one_is_uncontrolled :=TRUE;
        END_IF ;
    END_REPEAT ;
    are_aggregates := TRUE ;

    REPEAT i :=LOINDEX(elements) TO HIINDEX (elements) ;
    IF
        ('ISO13584_DICTIONARY_AGGREGATE_VALUE_SCHEMA.'+
        'AGGREGATE_ENTITY_INSTANCE_VALUE')
        IN TYPEOF(elements[i])
    THEN
        are_aggregates :=FALSE;
    END_IF ;
    END_REPEAT ;

    IF at_least_one_is_uncontrolled AND NOT(are_aggregates)

```

```

        THEN
            RETURN(UNKNOWN) ;
        END_IF ;
        REPEAT i :=LOINDEX(elements) TO HIINDEX(elements) ;
            IF NOT ((SIZEOF(type_of_elements.type_name)<>0 )
                AND
                (type_of_elements.type_name<= TYPEOF(elements[i])))
            THEN
                RETURN(FALSE) ;
            END_IF ;
        END_REPEAT ;
    END_IF ;

-- Aggregate of class instances
    IF ('ISO13584_IEC61630_DICTIONARY_SCHEMA.CLASS_INSTANCE_TYPE'
        IN TYPEOF(type_of_elements))
    THEN
        REPEAT i :=LOINDEX(elements) TO HIINDEX(elements) ;
            IF NOT((`ISO13584_INSTANCE_RESOURCE_SCHEMA.' +
                'DIC_CLASS_INSTANCE_VALUE' IN TYPEOF(elements[i]))
                AND compatible_class_and_class(type_of_elements.domain,
                    elements[i].class_def))
            THEN RETURN(FALSE) ;
            END_IF ;
        END_REPEAT ;
    END_IF ;

-- Aggregate of level values
    IF ('ISO13584_IEC61630_DICTIONARY_SCHEMA.LEVEL_TYPE'
        IN TYPEOF(type_of_elements))
    THEN
        REPEAT i :=LOINDEX(elements) TO HIINDEX(elements) ;
            IF NOT((`ISO13584_INSTANCE_RESOURCE_SCHEMA.' +
                'LEVEL_SPEC_VALUE' IN TYPEOF(elements[i]))
                AND compatible_level_type_and_instance(
                    type_of_elements.levels,
                    TYPEOF(type_of_elements\level_type.value_type),
                    elements[i]) )
            THEN RETURN(FALSE) ;
            END_IF ;
        END_REPEAT ;
    END_IF ;

-- Aggregate of named types values
    IF ('ISO13584_IEC61630_DICTIONARY_SCHEMA.NAMED_TYPE'
        IN TYPEOF(type_of_elements))
    THEN
        Result := TRUE ;
        REPEAT i :=LOINDEX(elements) TO HIINDEX(elements) ;
            Result := Result AND
                compatible_complete_types_and_value(
                    type_of_elements.Referred_type,

```

```

        elements[i] ) ;
    END_REPEAT ;
    RETURN (Result) ;
END_IF ;

-- check that values are themselves aggregates

-- Type of elements are aggregates
IF 'ISO13584_IEC61360_DICTIONARY.Aggregate_EXTENSION_SCHEMA.' +
    'AGREGATE_TYPE' IN TYPEOF(type_of_elements)
THEN
    IF 'ISO13584_IEC61360_DICTIONARY.Aggregate_EXTENSION_SCHEMA.' +
        'LIST_TYPE' IN TYPEOF(type_of_elements)
    THEN
        Aggregates_are_ok :=TRUE ;
        REPEAT i :=LOINDEX(elements) TO HIINDEX(elements) ;
            Aggregates_are_ok := aggregates_are_ok AND
                ('ISO13584_DICTIONARY.Aggregate_VALUE_SCHEMA.' +
                    'LIST_VALUE') IN TYPEOF(elements[i]) ;
        END_REPEAT ;
        IF NOT Aggregates_are_ok THEN RETURN (FALSE) ; END_IF ;
    END_IF ;

    IF 'ISO13584_IEC61360_DICTIONARY.Aggregate_EXTENSION_SCHEMA.' +
        'BAG_TYPE' IN TYPEOF(type_of_elements)
    THEN
        Aggregates_are_ok :=TRUE ;
        REPEAT i :=LOINDEX(elements) TO HIINDEX(elements) ;
            Aggregates_are_ok := aggregates_are_ok AND
                ('ISO13584_DICTIONARY.Aggregate_VALUE_SCHEMA.' +
                    'BAG_VALUE') IN TYPEOF(elements[i]) ;
        END_REPEAT ;
        IF NOT Aggregates_are_ok THEN RETURN (FALSE) ; END_IF ;
    END_IF ;

    IF 'ISO13584_IEC61360_DICTIONARY.Aggregate_EXTENSION_SCHEMA.' +
        'SET_TYPE' IN TYPEOF(type_of_elements)
    THEN
        Aggregates_are_ok :=TRUE ;
        REPEAT i :=LOINDEX(elements) TO HIINDEX(elements) ;
            Aggregates_are_ok := aggregates_are_ok AND
                ('ISO13584_DICTIONARY.Aggregate_VALUE_SCHEMA.' +
                    'SET_VALUE') IN TYPEOF(elements[i]) ;
        END_REPEAT ;
        IF NOT Aggregates_are_ok THEN RETURN (FALSE) ; END_IF ;
    END_IF ;

    IF 'ISO13584_IEC61360_DICTIONARY.Aggregate_EXTENSION_SCHEMA.' +
        'ARRAY_TYPE' IN TYPEOF(type_of_elements)
    THEN
        Aggregates_are_ok :=TRUE ;
        REPEAT i :=LOINDEX(elements) TO HIINDEX(elements) ;
            Aggregates_are_ok := aggregates_are_ok AND

```

```

        ((`ISO13584_DICTIONARY.Aggregate_Value_SCHEMA.' +
          'ARRAY_VALUE') IN TYPEOF(elements[i])) ;
      END_REPEAT ;
      IF NOT Aggregates_are_ok THEN RETURN (FALSE) ; END_IF ;
      END_IF ;
      -- check that the elements are compatible
      Result :=TRUE ;
      REPEAT i :=LOINDEX(elements) TO HIINDEX(elements) ;
        Result :=Result AND compatible_aggregate_type_and_value(
          type_of_elements.value_type, elements[i]) ;
      END_REPEAT ;
      RETURN(Result) ;
      END_IF ;

      -- values do not belong to a known type
      RETURN(UNKNOWN) ;

    END_FUNCTION ;
  */

```

7.4.3.Compatible_complete_types_and_value function

The **compatible_complete_types_and_value** function completes the function **compatible_type_and_value** defined in ISO 13584-24: 2001 by checking all the data types defined by this extension. It adds the checking of the aggregate values with respect to their defined data type. When the value associated to a property is not an aggregate, then the **compatible_type_and_value** is called, otherwise, the **compatible_aggregate_type_and_value** function is called.

EXPRESS specification:

```

*)
FUNCTION compatible_complete_types_and_value (
  dom : property_or_data_type_BSU ;
  val : primitive_value)
  : LOGICAL ;

```



```

IF data_type_typeof(dom)=[ ] THEN RETURN(UNKNOWN) ; END_IF ;

-- checking that values are primitive values but are not aggregate
-- values.
IF   (`ISO13584_INSTANCE_RESOURCE_SCHEMA.PRIMITIVE_VALUE'
      IN TYPEOF(val))
      AND
      (NOT (`ISO13584_DICTIONARY.Aggregate_Value_SCHEMA.' +
        'AGGREGATE_ENTITY_INSTANCE_VALUE'
        IN TYPEOF(val)))
THEN
  RETURN(compatible_type_and_value(dom,val)) ;

```

```

END_IF ;

IF `ISO13584_DICTIONARY_AGGREGATE_VALUE_SCHEMA.' +
    'AGGREGATE_ENTITY_INSTANCE_VALUE'
    IN TYPEOF(val)
THEN
    RETURN(compatible_aggregate_type_and_value(dom, val)) ;
END_IF ;

```

-- neither primitive value nor aggregate value.

```
RETURN(UNKNOWN) ;
```

```
END_FUNCTION ;
```

```
(*
```

7.5. ISO13584_IEC61360_dictionary_aggregate_extension_schema rule definition

7.5.1. Allowed_aggregate_values rule

The **allowed_aggregate_values** rule ensures that any **property_value** has its value compatible with the data type of the property even when this data type is an aggregate data type.

EXPRESS specification:

```

*)
RULE Allowed_aggregate_values FOR (property_value) ;

WHERE

WR1 :QUERY (prop <* property_value |
    NOT(compatible_complete_types_and_value(
        prop.prop_def, prop.its_value))) = [ ] ;

END_RULE ;
(*

```

Formal propositions:

WR1: the **its_value** values associated to the **prop_def property_BSU** of a **primitive_value** are type compatible with the data type defined by the **prop_def property_BSU**.

```

*)
END_SCHEMA ; -- ISO13584_dictionary_aggregate_value_schema;
(*)
```

8. Library integrated information model 25

Conformance to the library integrated information model 25 includes satisfying the information requirements stated in the **ISO13584_25_liim_schema** schema presented in annex A, the

requirements of the implementation method(s) supported, the relevant requirements of the normative references and the support of the standard data defined in annex B.

An implementation shall support at least the following implementation method: ISO10303-21. Requirements with respect to implementation methods are specified in annex C.

The **ISO13584_25_liim_schema** schema provides for a number of options that may be supported by an implementation. These options have been grouped into conformance classes. Five conformance classes are defined. Conformance to the library integrated information model 25 requires, as a minimum, conformance to class 0. Options are defined by each class and may be selected by an implementation. These options are also defined by means of EXPRESS schemas whose scopes are a subset of the complete scope of the **ISO13584_25_liim_schema** schema. Conformance to a particular conformance class requires that all the entities, types and associated constraints defined as part of the class, be supported, together with the standard data associated with the class.

The numbering schema of the conformance classes is as follows:

- class 0: minimal **dictionary_elements** from the ISO/IEC common dictionary schema with aggregate types,
- class 1: **dictionary_elements** from the extended dictionary schema with functional models and functional views,

NOTE 2 The extended dictionary schema is defined by the ISO13584_extended_dictionary_schema documented in ISO 13584-24: 2001.

- class 2: **dictionary_elements elements** from the extended dictionary schema with functional models and functional views and aggregate types and values,
- class 3: **dictionary_elements** from the extended dictionary with functional models and functional views and explicit class extension for the classes in the library,
- class 4: **dictionary_elements** from the extended dictionary with functional models and functional views, aggregate types and values and explicit class extension for the classes in the library,

NOTE 2 The attribute values for the **external_file_protocol** entities that do not belong to the standard data defined in this part of ISO 13584 (annexes B) or to the standard data defined in one part of the view exchange protocol series of part of ISO 13584 are subject to prior agreement between the sender and the receiver. They are outside the scope of this International Standard.

NOTE 3 The only files that may be referenced as **http_files** in conformance classes '1' to '4' of library integrated information model 25 are files whose MIME type and subtype:

- either corresponding to specifications that are publicly available, or
- that are associated with public domain Internet-available readers.

Table 1 shows the supported capabilities of the different conformance classes of library integrated information model 25.

Table 2 — Conformance options of library integrated information model 25

Capabilities	Dictionary elements			Library specification (class extension)
Conformance class	Dictionary definitions of item classes	Dictionary definitions of item classes and representation	Aggregate structured properties	
0	x		x	
1	x	x		
2	x	x	x	
3	x	x		x
4	x	x	x	x

8.1. Conformance class requirements

8.1.1. Conformance class 0 : ISO13584_short_form_CC0_schema short listing

This clause specifies the EXPRESS schema that uses elements either from the integrated resource series of ISO 10303 or from the logical resource or description methodology series of parts of ISO 13584 to define the requirements of library integrated information model LIIM 25, in conformance class 0.

NOTE 1 The integrated resource series of ISO 10303 are part 10303-4x and 10303-1xx. The logical resource series of parts of ISO 13584 are ISO 13584-2x and the description methodology series of parts of ISO 13584 are ISO 13584-4x.

Requirements stated in the following **ISO13584_short_form_CC0_schema** schema apply exclusively to those items that are used from this schema.

The expanded EXPRESS listing corresponding to the **ISO13584_short_form_CC0_schema** schema is presented in annex A.

NOTE 2 Conformance classes are defined by EXPRESS schemas for convenience only. The actual information model, and the name of the EXPRESS schema are defined by the **ISO13584_25_liim_schema** schema.

Conformance class 0 addresses those implementations that are intended to support the common requirements stated in the ISO/IEC dictionary schema and its extension which handles aggregate data types and values. An implementation of conformance class 0 of library integrated information model 25 shall support the entities and related constructs described in the following EXPRESS schema.

EXPRESS specification:

*)

```

SCHEMA ISO13584_short_form_CC0_schema;

USE FROM ISO13584_IEC61360_dictionary_schema (
    supplier_BSU,
    supplier_element,
    class_BSU,
    item_class,
    component_class,
    material_class,
    property_BSU,
    property_DET,
```

```

condition_DET,
dependent_P_DET
non_dependent_P_DET,
class_value_assignment,
data_type_BSU,
data_type_element,
number_type,
int_type,
int_measure_type,
int_currency_type,
non_quantitative_int_type,
real_type,
real_measure_type,
real_currency_type,
boolean_type,
string_type,
non_quantitative_code_type,
complex_type,
level_type,
class_instance_type,
entity_instance_type,
placement_type,
axis1_placement_type,
axis2_placement_2d_type,
axis2_placement_3d_type,
named_type,
value_domain,
dic_value,
non_si_unit,
dic_unit,
dates,
identified_document,
item_names,
label_with_language,
mathematical_string
) ;

USE FROM ISO13584_IEC61360_language_resource_schema (
global_language_assignment,
present_translations,
translated_label,
translated_text
) ;

USE FROM ISO13584_IEC61360_dictionary_aggregate_extension_schema (
aggregate_entity_instance_type,
list_type,
set_type,
bag_type,
array_type
) ;

```

```

USE FROM measure_schema (
    amount_of_substance_measure,
    amount_of_substance_measure_with_unit,
    amount_of_substance_unit,
    area_measure,
    area_measure_with_unit,
    area_unit,
    context_dependent_measure,
    context_dependent_unit,
    conversion_based_unit,
    count_measure,
    derived_unit,
    derived_unit_element,
    descriptive_measure,
    dimensional_exponents,
    electric_current_measure,
    electric_current_measure_with_unit,
    electric_current_unit,
    global_unit_assigned_context,
    length_measure,
    length_measure_with_unit,
    length_unit,
    luminous_intensity_measure,
    luminous_intensity_measure_with_unit,
    luminous_intensity_unit,
    mass_measure,
    mass_measure_with_unit,
    mass_unit,
    measure_value,
    measure_with_unit,
    named_unit,
    numeric_measure,
    parameter_value,
    plane_angle_measure,
    plane_angle_measure_with_unit,
    plane_angle_unit,
    positive_length_measure,
    positive_plane_angle_measure,
    positive_ratio_measure,
    ratio_measure,
    ratio_measure_with_unit,
    ratio_unit,
    si_unit,
    solid_angle_measure,
    solid_angle_measure_with_unit,
    solid_angle_unit,
    thermodynamic_temperature_measure,
    thermodynamic_temperature_measure_with_unit,
    thermodynamic_temperature_unit,
    time_measure,
    time_measure_with_unit,
    time_unit,
    volume_measure,
)

```

```

        volume_measure_with_unit,
        volume_unit
    ) ;

USE FROM person_organization_schema (
    address,
    organization,
    person
) ;

USE FROM geometry_schema (
    axis1_placement,
    axis2_placement_2D,
    axis2_placement_3D,
    geometric_representation_context,
    placement
) ;
(*

```

8.1.1.1 ISO13584_short_form_CC0_schema global rule definitions

The following rules define the requirements for the **ISO13584_short_form_CC0_schema** schema.

8.1.1.1.1 No_content_without_DET_rule rule

The **no_content_without_DET_rule** states that no **content_item** of any **basic_semantic_unit** shall be provided if its **dictionary_element** definition is not provided in the same exchange context. Its informal proposition states that when a **dictionary_element** is provided, the **content_item** that refers to the same **basic_semantic_unit** shall be provided if it exists. This rule ensures that all **content_items** that might contribute, to the definition of a class shall be available in an exchange context if all the **dictionary_elements** that describe this class and all its superclass are available in this exchange context.

EXPRESS specification:

```

*)
RULE no_content_without_DET_rule FOR (basic_semantic_unit);
WHERE
    WR1: QUERY (bsu <* basic_semantic_unit
        | (SIZEOF (bsu.referenced_by) = 1)
        AND (SIZEOF (bsu.definition) = 0)) = [] ;
END_RULE;
(*

```

Formal proposition:

WR1: no **content_item** of any **basic_semantic_unit** shall be provided if its **dictionary_element** definition is not provided in the same exchange context.

Informal proposition:

IP1: when a **dictionary_element** is provided, the **content_item** that refers to the same **basic_semantic_unit** shall be provided if it exists.

```
*)
END_SCHEMA; -- ISO13584_short_form_CC0_schema
```

```
(*
```

8.1.2. Conformance class 1 : ISO13584_short_form_CC1_schema short listing

This clause specifies the EXPRESS schema that uses elements either from the integrated resource series of ISO 10303 or from the logical resource or description methodology series of parts of ISO 13584 to define the requirements of library integrated information model LIIM 25 in conformance class 1.

NOTE 1 The integrated resource series of ISO 10303 are part 10303-4x and 10303-1xx. The logical resource series of parts of ISO 13584 are ISO 13584-2x and the description methodology series of parts of ISO 13584 are ISO 13584-4x.

Requirements stated in the following **ISO13584_short_form_CC1_schema** schema apply exclusively to those items that are used from this schema.

The expanded EXPRESS listing corresponding to the **ISO13584_short_form_CC1_schema** schema is presented in annex A.

NOTE 2 Conformance classes are defined by EXPRESS schemas for convenience only. The actual information model, and the name of the EXPRESS schema are defined by the **ISO13584_25_liim_schema** schema.

Conformance class 1 addresses those implementations that are intended to support the common requirements stated in the ISO/IEC dictionary and in the **ISO13584_extended_dictionary_schema** schema. An implementation of conformance class 1 of library integrated information model 25 shall support the entities and related constructs described in the following EXPRESS schema.

EXPRESS specification:

```
*)
SCHEMA ISO13584_short_form_CC1_schema;

USE FROM ISO13584_IEC61360_dictionary_schema (
    supplier_BSU,
    supplier_element,
    class_BSU,
    item_class,
    component_class,
    material_class,
    property_BSU,
    property_DET,
    condition_DET,
    dependent_P_DET
    non_dependent_P_DET,
    class_value_assignment,
    data_type_BSU,
    data_type_element,
    number_type,
    int_type,
    int_measure_type,
    int_currency_type,
    non_quantitative_int_type,
    real_type,
    real_measure_type,
    real_currency_type,
    boolean_type,
```

```

string_type,
non_quantitative_code_type,
complex_type,
level_type,
class_instance_type,
entity_instance_type,
placement_type,
axis1_placement_type,
axis2_placement_2d_type,
axis2_placement_3d_type,
named_type,
value_domain,
dic_value,
non_si_unit,
dic_unit,
dates,
identified_document,
item_names,
label_with_language,
mathematical_string
);

USE FROM ISO13584_IEC61360_language_resource_schema (
    global_language_assignment,
    present_translations,
    translated_label,
    translated_text
);

USE FROM ISO13584_extended_dictionary_schema (
    dictionary,
    dictionary_in_standard_format,
    library_iim_id,
    view_exchange_protocol_id,
    representation_type,
    geometric_representation_context_type,
    representation_reference_type,
    document_BSU,
    supplier_program_library_relationship,
    class_document_relationship,
    representation_P_DET,
    class_related_element,
    document_element,
    document_element_with_http_access,
    documented_element_with_translated_http_access,
    referenced_document,
    referenced_graphics,
    feature_class,
    functional_model_class,
    fm_class_view_of,
    functional_view_class,
    non_instantiable_functional_view_class,
);

```

```

view_control_variable_range,
item_class_case_of,
component_class_case_of,
material_class_case_of,
feature_class_case_of,
a_posteriori_case_of,
a_posteriori_view_of,
);

USE FROM ISO13584_external_file_schema (
    standard_data_protocol,
    non_standard_data_protocol,
    http_protocol,
    document_content,
    message,
    illustration,
    A6_illustration,
    A9_illustration,
    translated_external_content,
    not_translated_external_content,
    not_translatable_external_content,
    language_specific_content,
    external_file_unit,
    http_file,
    http_class_directory
);

USE FROM measure_schema (
    amount_of_substance_measure,
    amount_of_substance_measure_with_unit,
    amount_of_substance_unit,
    area_measure,
    area_measure_with_unit,
    area_unit,
    context_dependent_measure,
    context_dependent_unit,
    conversion_based_unit,
    count_measure,
    derived_unit,
    derived_unit_element,
    descriptive_measure,
    dimensional_exponents,
    electric_current_measure,
    electric_current_measure_with_unit,
    electric_current_unit,
    global_unit_assigned_context,
    length_measure,
    length_measure_with_unit,
    length_unit,
    luminous_intensity_measure,
    luminous_intensity_measure_with_unit,
    luminous_intensity_unit,
    mass_measure,
);

```

```

mass_measure_with_unit,
mass_unit,
measure_value,
measure_with_unit,
named_unit,
numeric_measure,
parameter_value,
plane_angle_measure,
plane_angle_measure_with_unit,
plane_angle_unit,
positive_length_measure,
positive_plane_angle_measure,
positive_ratio_measure,
ratio_measure,
ratio_measure_with_unit,
ratio_unit,
si_unit,
solid_angle_measure,
solid_angle_measure_with_unit,
solid_angle_unit,
thermodynamic_temperature_measure,
thermodynamic_temperature_measure_with_unit,
thermodynamic_temperature_unit,
time_measure,
time_measure_with_unit,
time_unit,
volume_measure,
volume_measure_with_unit,
volume_unit
);

USE FROM person_organization_schema (
    address,
    organization,
    person
);

USE FROM geometry_schema (
    axis1_placement,
    axis2_placement_2D,
    axis2_placement_3D,
    geometric_representation_context,
    placement
);

USE FROM representation_schema (
    representation,
    representation_context,
    representation_item
);

USE FROM application_context_schema(

```

```

    application_context,
    application_context_element,
    application_protocol_definition
);
(*

```

8.1.2.1 ISO13584_short_form_CC1_schema global rule definitions

The following rules define the requirements for the **ISO13584_short_form_CC1_schema** schema.

8.1.2.1.1 No_content_without_DET_rule rule

The **no_content_without_DET_rule** states that no **content_item** of any **basic_semantic_unit** shall be provided if its **dictionary_element** definition is not provided in the same exchange context. Its informal proposition states that when a **dictionary_element** is provided, the **content_item** that refers to the same **basic_semantic_unit** shall be provided if it exists. This rule ensures that all **content_items** that might contribute, through inheritance, to the definition of a class shall be available in an exchange context if all the **dictionary_elements** that describe this class and all its superclass are available in this exchange context.

EXPRESS specification:

```

*)
RULE no_content_without_DET_rule FOR (basic_semantic_unit);
WHERE
    WR1: QUERY (bsu <* basic_semantic_unit
        | (SIZEOF (bsu.referenced_by) = 1)
        AND (SIZEOF (bsu.definition) = 0)) = [];
END_RULE;
(*

```

Formal proposition:

WR1: no **content_item** of any **basic_semantic_unit** shall be provided if its **dictionary_element** definition is not provided in the same exchange context.

Informal proposition:

IP1: when a **dictionary_element** is provided, the **content_item** that refers to the same **basic_semantic_unit** shall be provided if it exists.

```

*)
END_SCHEMA; -- ISO13584_short_form_CC1_schema
(*)
```

8.1.3.Conformance class 2 : ISO13584_short_form_CC2_schema short listing

This clause specifies the EXPRESS schema that uses elements either from the integrated resource series of ISO 10303 or from the logical resource or description methodology series of parts of ISO 13584 to define the requirements of library integrated information model LIIM 25, in conformance class 2.

NOTE 1 The integrated resource series of ISO 10303 are part 10303-4x and 10303-1xx. The logical resource series of parts of ISO 13584 are ISO 13584-2x and the description methodology series of parts of ISO 13584 are ISO 13584-4x.

Requirements stated in the following **ISO13584_short_form_CC2_schema** schema apply exclusively to those items that are used from this schema.

The expanded EXPRESS listing corresponding to the **ISO13584_short_form_CC2_schema** schema is presented in annex A.

NOTE 2 Conformance classes are defined by EXPRESS schemas for convenience only. The actual information model, and the name of the EXPRESS schema are defined by the **ISO13584_25_liim_schema** schema.

Conformance class 2 addresses those implementations that support conformance class 1 and may support aggregate data types and values.

EXPRESS specification:

*)

```
SCHEMA ISO13584_short_form_CC2_schema;

USE FROM ISO13584_IEC61360_dictionary_schema(
    supplier_BSU,
    supplier_element,
    class_BSU,
    item_class,
    component_class,
    material_class,
    property_BSU,
    property_DET,
    condition_DET,
    dependent_P_DET
    non_dependent_P_DET,
    class_value_assignment,
    data_type_BSU,
    data_type_element,
    number_type,
    int_type,
    int_measure_type,
    int_currency_type,
    non_quantitative_int_type,
    real_type,
    real_measure_type,
    real_currency_type,
    boolean_type,
    string_type,
    non_quantitative_code_type,
    complex_type,
    level_type,
    class_instance_type,
    entity_instance_type,
    placement_type,
    axis1_placement_type,
    axis2_placement_2d_type,
    axis2_placement_3d_type,
    named_type,
    value_domain,
    dic_value,
```

```

non_si_unit,
dic_unit,
dates,
identified_document,
item_names,
label_with_language,
mathematical_string
);

USE FROM ISO13584_IEC61360_language_resource_schema (
global_language_assignment,
present_translations,
translated_label,
translated_text
);

USE FROM ISO13584_IEC61360_dictionary_aggregate_extension_schema (
aggregate_entity_instance_type,
list_type,
set_type,
bag_type,
array_type
);

USE FROM ISO13584_extended_dictionary_schema (
dictionary,
dictionary_in_standard_format,
library_iim_id,
view_exchange_protocol_id,
representation_type,
geometric_representation_context_type,
representation_reference_type,
document_BSU,
supplier_program_library_relationship,
class_document_relationship,
representation_P_DET,
class_related_element,
document_element,
document_element_with_http_access,
documented_element_with_translated_http_access,
referenced_document,
referenced_graphics,
feature_class,
functional_model_class,
fm_class_view_of,
functional_view_class,
non_instantiable_functional_view_class,
view_control_variable_range,
item_class_case_of,
component_class_case_of,
material_class_case_of,
feature_class_case_of,
a_posteriori_case_of,

```

```
a_posteriori_view_of,  
);  
  
USE FROM ISO13584_external_file_schema (   
standard_data_protocol,  
non_standard_data_protocol,  
http_protocol,  
document_content,  
message,  
illustration,  
A6_illustration,  
A9_illustration,  
translated_external_content,  
not_translated_external_content,  
not_translatable_external_content,  
language_specific_content,  
external_file_unit,  
http_file,  
http_class_directory  
);  
  
USE FROM measure_schema (   
amount_of_substance_measure,  
amount_of_substance_measure_with_unit,  
amount_of_substance_unit,  
area_measure,  
area_measure_with_unit,  
area_unit,  
context_dependent_measure,  
context_dependent_unit,  
conversion_based_unit,  
count_measure,  
derived_unit,  
derived_unit_element,  
descriptive_measure,  
dimensional_exponents,  
electric_current_measure,  
electric_current_measure_with_unit,  
electric_current_unit,  
global_unit_assigned_context,  
length_measure,  
length_measure_with_unit,  
length_unit,  
luminous_intensity_measure,  
luminous_intensity_measure_with_unit,  
luminous_intensity_unit,  
mass_measure,  
mass_measure_with_unit,  
mass_unit,  
measure_value,  
measure_with_unit,  
named_unit,
```

```

    numeric_measure,
    parameter_value,
    plane_angle_measure,
    plane_angle_measure_with_unit,
    plane_angle_unit,
    positive_length_measure,
    positive_plane_angle_measure,
    positive_ratio_measure,
    ratio_measure,
    ratio_measure_with_unit,
    ratio_unit,
    si_unit,
    solid_angle_measure,
    solid_angle_measure_with_unit,
    solid_angle_unit,
    thermodynamic_temperature_measure,
    thermodynamic_temperature_measure_with_unit,
    thermodynamic_temperature_unit,
    time_measure,
    time_measure_with_unit,
    time_unit,
    volume_measure,
    volume_measure_with_unit,
    volume_unit
);

USE FROM person_organization_schema (
    address,
    organization,
    person
);

USE FROM geometry_schema (
    axis1_placement,
    axis2_placement_2D,
    axis2_placement_3D,
    geometric_representation_context,
    placement
);

USE FROM representation_schema (
    representation,
    representation_context,
    representation_item
);

USE FROM application_context_schema (
    application_context,
    application_context_element,
    application_protocol_definition
);
(*

```

8.1.3.1 ISO13584_short_form_CC2_schema global rule definitions

The following rules define the requirements for the **ISO13584_short_form_CC2_schema** schema.

8.1.3.1.1 No_content_without_DET_rule rule

The **no_content_without_DET_rule** states that no **content_item** of any **basic_semantic_unit** shall be provided if its **dictionary_element** definition is not provided in the same exchange context. Its informal proposition states that when a **dictionary_element** is provided, the **content_item** that refers to the same **basic_semantic_unit** shall be provided if it exists. This rule ensures that all **content_items** that might contribute, through inheritance, to the definition of a class shall be available in an exchange context if all the **dictionary_elements** that describe this class and all its superclass are available in this exchange context.

EXPRESS specification:

```

*)
RULE no_content_without_DET_rule FOR (basic_semantic_unit);
WHERE
    WR1: QUERY (bsu <* basic_semantic_unit
        | (SIZEOF (bsu.referenced_by) = 1)
        AND (SIZEOF (bsu.definition) = 0)) = [] ;
END_RULE;
(*

```

Formal proposition:

WR1: no **content_item** of any **basic_semantic_unit** shall be provided if its **dictionary_element** definition is not provided in the same exchange context.

Informal proposition:

IP1: when a **dictionary_element** is provided, the **content_item** that refers to the same **basic_semantic_unit** shall be provided if it exists.

```

*)
END_SCHEMA; -- ISO13584_short_form_CC2_schema
(*)
```

8.1.4. Conformance class 3 : ISO13584_short_form_CC3_schema short listing

This clause specifies the EXPRESS schema that uses elements either from the integrated resource series of ISO 10303 or from the logical resource or description methodology series of parts of ISO 13584 to define the requirements of library integrated information model LIIM 25, in conformance class 3.

NOTE 1 The integrated resource series of ISO 10303 are part 10303-4x and 10303-1xx. The logical resource series of parts of ISO 13584 are ISO 13584-2x and the description methodology series of parts of ISO 13584 are ISO 13584-4x.

Requirements stated in the following **ISO13584_short_form_CC3_schema** schema apply exclusively to those items that are used from this schema.

The expanded EXPRESS listing corresponding to the **ISO13584_short_form_CC3_schema** schema is presented in annex A.

NOTE 2 Conformance classes are defined by EXPRESS schemas for convenience only. The actual information model, and the name of the EXPRESS schema are defined by the **ISO13584_25_liim_schema** schema.

Conformance class 3 addresses those implementations that support conformance class 1 and explicit description of class extensions by means of the definition of their set of instances.

EXPRESS specification:

```

*)  

SCHEMA ISO13584_short_form_CC3_schema;  

USE FROM ISO13584_IEC61360_dictionary_schema (
    supplier_BSU,
    supplier_element,
    class_BSU,
    item_class,
    component_class,
    material_class,
    property_BSU,
    property_DET,
    condition_DET,
    dependent_P_DET
    non_dependent_P_DET,
    class_value_assignment,
    data_type_BSU,
    data_type_element,
    number_type,
    int_type,
    int_measure_type,
    int_currency_type,
    non_quantitative_int_type,
    real_type,
    real_measure_type,
    real_currency_type,
    boolean_type,
    string_type,
    non_quantitative_code_type,
    complex_type,
    level_type,
    class_instance_type,
    entity_instance_type,
    placement_type,
    axis1_placement_type,
    axis2_placement_2d_type,
    axis2_placement_3d_type,
    named_type,
    value_domain,
    dic_value,
    non_si_unit,
    dic_unit,
    dates,
    identified_document,
    item_names,
    label_with_language,
)

```

```

    mathematical_string
) ;

USE FROM ISO13584_IEC61360_language_resource_schema (
    global_language_assignment,
    present_translations,
    translated_label,
    translated_text
) ;

USE FROM ISO13584_extended_dictionary_schema (
    dictionary,
    dictionary_in_standard_format,
    library_iim_id,
    view_exchange_protocol_id,
    representation_type,
    geometric_representation_context_type,
    representation_reference_type,
    document_BSU,
    supplier_program_library_relationship,
    class_document_relationship,
    representation_P_DET,
    class_related_element,
    document_element,
    document_element_with_http_access,
    documented_element_with_translated_http_access,
    referenced_document,
    referenced_graphics,
    feature_class,
    functional_model_class,
    fm_class_view_of,
    functional_view_class,
    non_instantiable_functional_view_class,
    view_control_variable_range,
    item_class_case_of,
    component_class_case_of,
    material_class_case_of,
    feature_class_case_of,
    a_posteriori_case_of,
    a_posteriori_view_of,
) ;

USE FROM ISO13584_external_file_schema (
    standard_data_protocol,
    non_standard_data_protocol,
    http_protocol,
    document_content,
    message,
    illustration,
    A6_illustration,
    A9_illustration,
    translated_external_content,
)

```

```

not_translated_external_content,
not_translatable_external_content,
language_specific_content,
external_file_unit,
http_file,
http_class_directory
);

USE FROM ISO13584_instance_resource_schema (
level_spec_value,
    int_level_spec_value,
    real_level_spec_value,
    dic_component_instance,
    dic_material_instance,
    dic_feature_instance,
    lib_item_instance,
    lib_component_instance,
    lib_material_instance,
    lib_feature_instance,
    dic_f_model_instance,
    lib_f_model_instance,
    dic_f_view_instance,
    property_value,
    context_dependent_property_value
);

USE FROM ISO13584_library_content_schema (
library,
    library_in_standard_format,
    explicit_item_class_extension,
    explicit_functional_model_class_extension
);

USE FROM measure_schema (
amount_of_substance_measure,
amount_of_substance_measure_with_unit,
amount_of_substance_unit,
area_measure,
area_measure_with_unit,
area_unit,
context_dependent_measure,
context_dependent_unit,
conversion_based_unit,
count_measure,
derived_unit,
derived_unit_element,
descriptive_measure,
dimensional_exponents,
electric_current_measure,
electric_current_measure_with_unit,
electric_current_unit,
global_unit_assigned_context,
length_measure,

```

```

length_measure_with_unit,
length_unit,
luminous_intensity_measure,
luminous_intensity_measure_with_unit,
luminous_intensity_unit,
mass_measure,
mass_measure_with_unit,
mass_unit,
measure_value,
measure_with_unit,
named_unit,
numeric_measure,
parameter_value,
plane_angle_measure,
plane_angle_measure_with_unit,
plane_angle_unit,
positive_length_measure,
positive_plane_angle_measure,
positive_ratio_measure,
ratio_measure,
ratio_measure_with_unit,
ratio_unit,
si_unit,
solid_angle_measure,
solid_angle_measure_with_unit,
solid_angle_unit,
thermodynamic_temperature_measure,
thermodynamic_temperature_measure_with_unit,
thermodynamic_temperature_unit,
time_measure,
time_measure_with_unit,
time_unit,
volume_measure,
volume_measure_with_unit,
volume_unit
);

USE FROM person_organization_schema (
    address,
    organization,
    person
);

USE FROM geometry_schema (
    axis1_placement,
    axis2_placement_2D,
    axis2_placement_3D,
    geometric_representation_context,
    placement
);

USE FROM representation_schema (

```

```

representation,
representation_context,
representation_item
);

USE FROM application_context_schema (
    application_context,
    application_context_element,
    application_protocol_definition
);
(*

```

8.1.4.1 ISO13584_short_form_CC3_schema global rule definitions

The following rules define the requirements for the **ISO13584_short_form_CC4_schema** schema.

8.1.4.1.1 No_content_without_DET_rule rule

The **no_content_without_DET_rule** states that no **content_item** of any **basic_semantic_unit** shall be provided if its **dictionary_element** definition is not provided in the same exchange context. Its informal proposition states that when a **dictionary_element** is provided, the **content_item** that refers to the same **basic_semantic_unit** shall be provided if it exists. This rule ensures that all **content_items** that might contribute, through inheritance, to the definition of a class shall be available in an exchange context if all the **dictionary_elements** that describe this class and all its superclass are available in this exchange context.

EXPRESS specification:

```

*)
RULE no_content_without_DET_rule FOR (basic_semantic_unit);
WHERE
    WR1: QUERY (bsu <* basic_semantic_unit
        | (SIZEOF (bsu.referenced_by) = 1)
        AND (SIZEOF (bsu.definition) = 0)) = [];
END_RULE;
(*

```

Formal proposition:

WR1: no **content_item** of any **basic_semantic_unit** shall be provided if its **dictionary_element** definition is not provided in the same exchange context.

Informal proposition:

IP1: when a **dictionary_element** is provided, the **content_item** that refers to the same **basic_semantic_unit** shall be provided if it exists.

```

*)
END_SCHEMA; -- ISO13584_short_form_CC3_schema
(*)
```

8.1.5. Conformance class 4 : ISO13584_25_liim_schema short listing

This clause specifies the EXPRESS schema that uses elements either from the integrated resource series of ISO 10303 or from the logical resource or description methodology series of parts of ISO

13584 to define the requirements of library integrated information model LIIM 25, in conformance class 4.

NOTE 1 The integrated resource series of ISO 10303 are part 10303-4x and 10303-1xx. The logical resource series of parts of ISO 13584 are ISO 13584-2x and the description methodology series of parts of ISO 13584 are ISO 13584-4x.

Requirements stated in the following **ISO13584_25_liim_schema** schema apply exclusively to those items that are used from this schema.

The expanded EXPRESS listing corresponding to the **ISO13584_25_liim_schema** schema is presented in annex A.

NOTE 2 Conformance classes are defined by EXPRESS schemas for convenience only. The actual information model, and the name of the EXPRESS schema are defined by the **ISO13584_25_liim_schema** schema.

Conformance class 4 addresses those implementations that support conformance class 2 and explicit description of class extensions by means of the definition of their set of instances.

EXPRESS specification:

*)

```
SCHEMA ISO13584_short_form_CC4_schema;

USE FR OM ISO13584_IEC61360_dictionary_schema(
    supplier_BSU,
    supplier_element,
    class_BSU,
    item_class,
    component_class,
    material_class,
    property_BSU,
    property_DET,
    condition_DET,
    dependent_P_DET
    non_dependent_P_DET,
    class_value_assignment,
    data_type_BSU,
    data_type_element,
    number_type,
    int_type,
    int_measure_type,
    int_currency_type,
    non_quantitative_int_type,
    real_type,
    real_measure_type,
    real_currency_type,
    boolean_type,
    string_type,
    non_quantitative_code_type,
    complex_type,
    level_type,
    class_instance_type,
    entity_instance_type,
```

```

placement_type,
axis1_placement_type,
axis2_placement_2d_type,
axis2_placement_3d_type,
named_type,
value_domain,
dic_value,
non_si_unit,
dic_unit,
dates,
identified_document,
item_names,
label_with_language,
mathematical_string
);

USE FROM ISO13584_IEC61360_language_resource_schema (
    global_language_assignment,
    present_translations,
    translated_label,
    translated_text
);

USE FROM ISO13584_IEC61360_dictionary_aggregate_extension_schema (
    aggregate_entity_instance_type,
    list_type,
    set_type,
    bag_type,
    array_type
);

USE FROM ISO13584_extended_dictionary_schema (
    dictionary,
    dictionary_in_standard_format,
    library_iim_id,
    view_exchange_protocol_id,
    representation_type,
    geometric_representation_context_type,
    representation_reference_type,
    document_BSU,
    supplier_program_library_relationship,
    class_document_relationship,
    representation_P_DET,
    class_related_element,
    document_element,
    document_element_with_http_access,
    documented_element_with_translated_http_access,
    referenced_document,
    referenced_graphics,
    feature_class,
    functional_model_class,
    fm_class_view_of,
    functional_view_class,
);

```

```

non_instantiable_functional_view_class,
view_control_variable_range,
item_class_case_of,
component_class_case_of,
material_class_case_of,
feature_class_case_of,
a_posteriori_case_of,
a_posteriori_view_of,
) ;

USE FROM ISO13584_external_file_schema(
    standard_data_protocol,
    non_standard_data_protocol,
    http_protocol,
    document_content,
    message,
    illustration,
    A6_illustration,
    A9_illustration,
    translated_external_content,
    not_translated_external_content,
    not_translatable_external_content,
    language_specific_content,
    external_file_unit,
    http_file,
    http_class_directory
) ;

USE FROM ISO13584_instance_resource_schema(
    level_spec_value,
    int_level_spec_value,
    real_level_spec_value,
    dic_component_instance,
    dic_material_instance,
    dic_feature_instance,
    lib_item_instance,
    lib_component_instance,
    lib_material_instance,
    lib_feature_instance,
    dic_f_model_instance,
    lib_f_model_instance,
    dic_f_view_instance,
    property_value,
    context_dependent_property_value
) ;

USE FROM ISO13584_dictionary_aggregate_value_schema(
    aggregate_entity_instance_value,
    list_value,
    set_value,
    bag_value,
    array_value
)

```

```

) ;

USE FROM ISO13584_library_content_schema (
    library,
    library_in_standard_format,
    explicit_item_class_extension,
    explicit_functional_model_class_extension
) ;

USE FROM measure_schema (
    amount_of_substance_measure,
    amount_of_substance_measure_with_unit,
    amount_of_substance_unit,
    area_measure,
    area_measure_with_unit,
    area_unit,
    context_dependent_measure,
    context_dependent_unit,
    conversion_based_unit,
    count_measure,
    derived_unit,
    derived_unit_element,
    descriptive_measure,
    dimensional_exponents,
    electric_current_measure,
    electric_current_measure_with_unit,
    electric_current_unit,
    global_unit_assigned_context,
    length_measure,
    length_measure_with_unit,
    length_unit,
    luminous_intensity_measure,
    luminous_intensity_measure_with_unit,
    luminous_intensity_unit,
    mass_measure,
    mass_measure_with_unit,
    mass_unit,
    measure_value,
    measure_with_unit,
    named_unit,
    numeric_measure,
    parameter_value,
    plane_angle_measure,
    plane_angle_measure_with_unit,
    plane_angle_unit,
    positive_length_measure,
    positive_plane_angle_measure,
    positive_ratio_measure,
    ratio_measure,
    ratio_measure_with_unit,
    ratio_unit,
    si_unit,
    solid_angle_measure,
)

```

```

solid_angle_measure_with_unit,
solid_angle_unit,
thermodynamic_temperature_measure,
thermodynamic_temperature_measure_with_unit,
thermodynamic_temperature_unit,
time_measure,
time_measure_with_unit,
time_unit,
volume_measure,
volume_measure_with_unit,
volume_unit
);

USE FROM person_organization_schema (
    address,
    organization,
    person
);

USE FROM geometry_schema (
    axis1_placement,
    axis2_placement_2D,
    axis2_placement_3D,
    geometric_representation_context,
    placement
);

USE FROM representation_schema (
    representation,
    representation_context,
    representation_item
);

USE FROM application_context_schema (
    application_context,
    application_context_element,
    application_protocol_definition
);

```

(*
8.1.5.1 ISO13584_short_form_CC4_schema global rule definitions

The following rules define the requirements for the **ISO13584_short_form_CC4_schema** schema.

8.1.5.1.1 No_content_without_DET_rule rule

The **no_content_without_DET_rule** states that no **content_item** of any **basic_semantic_unit** shall be provided if its **dictionary_element** definition is not provided in the same exchange context. Its informal proposition states that when a **dictionary_element** is provided, the **content_item** that refers to the same **basic_semantic_unit** shall be provided if it exists. This rule ensures that all **content_items** that might contribute, through inheritance, to the definition of a class shall be available in an exchange context if all the **dictionary_elements** that describe this class and all its superclass are available in this exchange context.

EXPRESS specification:

```

*)  

RULE no_content_without_DET_rule FOR (basic_semantic_unit);  

WHERE  

    WR1: QUERY (bsu <* basic_semantic_unit  

        | (SIZEOF (bsu.referenced_by) = 1)  

        AND (SIZEOF (bsu.definition) = 0)) = [];  

END_RULE;  

(*

```

Formal proposition:

WR1: no **content_item** of any **basic_semantic_unit** shall be provided if its **dictionary_element** definition is not provided in the same exchange context.

Informal proposition:

IP1: when a **dictionary_element** is provided, the **content_item** that refers to the same **basic_semantic_unit** shall be provided if it exists.

```

*)  

END_SCHEMA; -- ISO13584_short_form_CC4_schema  

(*

```

Annex A (informative) ISO13584_25_liim_schema and conformance classes expanded listing

This annex references a listing of the complete EXPRESS schemas specified in Clause 8 of this part of ISO 13584 without comments or other explanatory text but with the additional constraints defined in annex B

In these listings, all the elements used either from the integrated resources of ISO 10303 (ISO 10303-4x) or from the logical resource or description methodology series of parts of ISO 13584 (ISO 13584-2x and ISO 13584-4x) and the constraints defined in the short forms specified in clause 8 and in the normative annex B gathered in an unique schema without any external reference. These schemas may be used to exchange libraries that does not reference any view exchange protocol, as specified in annex B of this part of ISO 13584. These listings are available in computer-interpretable form and can be found at the following URL:

<http://www.nist.gov/sc4/plib/part025/cd/>

If there is difficulty accessing these sites contact ISO Central Secretariat or contact the ISO TC 184/SC4 Secretariat directly at: sc4sec@cme.nist.gov.

NOTE 1 The information provided in computer-interpretable form at the above URLs is informative. The information that is contained in the body of this part of ISO 13584 is normative.

NOTE 2 If some errors are identified in the EXPRESS code during the ballot process, the description of these errors, together with the corrections recommended for PLIB implementations by the part editors can be found at the following URL:

http://www.lisi.ensma.fr/ftp/pub/PLIB_release_notes/Part25/Part25-WD/

Annex B (normative)

Standard data requirements for the library integrated information model 25

Standard data are the entity instances that shall be recognized by any implementation compliant with ISO 13584 that claims conformance to some conformance class of some library integrated information model or view exchange protocol of ISO 13584.

Standard data shall be specified by each library integrated information model and by each view exchange protocol, and for each of them, for each conformance class.

Standard data may include:

- instances of **basic_semantic_units**, associated with the corresponding **dictionary_element** and possibly **content_item**,
- instances of **external_file_protocols**, and
- instance of other entities required to define the previous entities instances.

Recognition of a **basic_semantic_unit** means that the corresponding **basic_semantic_unit** shall be already stored in the user system, together with a corresponding **dictionary_element** and possibly a **content_item** as specified in the view exchange protocol or library integrated information model standard data. This implies that a reference to a value-equal **basic_semantic_unit** in a supplier library is interpreted as a reference to the pre-existing **basic_semantic_unit**.

NOTE 1 Examples of **basic_semantic_units** that may be defined as standard data in a view exchange protocol include the **class_BSU** that identifies the functional view class possibly defined by the view exchange protocol and the **property_BSU** that identifies the view control variable of this functional view class.

Recognition of an external file protocol means that external files that reference a value equal **external_file_protocol** shall be processed by an implementation that recognize this **external_file_protocol**.

NOTE 2 Example of an external file protocol that may be defined as standard data by a view exchange protocol or a library integrated information model is the ISO standard ISO 8859-1 that specifies a 8-bit single byte coded graphics character set for Latin alphabet N°1.

Standard data are specified by means of a set of constraints that shall be fulfilled by any library that claims conformance to some conformance class of LIIM 25. The following standard data are specified by library integrated information model 25.

B.1 Constraints on a library delivery file for referencing library integrated information model 25

This subclause defines the **library_liim_id** instance values that are allowed for use in a library delivery file to reference library integrated information model 25 defined in this part of ISO 13584.

The set of allowed values is defined by means of Table B.1 that specifies for each conformance class the allowed values of **library_liim_id.name** and **library_liim_id.application**, and by means of one EXPRESS schema that contains a global rule. This rules shall be fulfilled by any library delivery file that references library integrated information model 25, defined in this part of ISO 13584 in any of its conformance class. The goal of this rule is to specify the allowed values for the other attributes of **library_liim_id** that shall be used to reference library integrated information model 25.

Formally, the information model of a library delivery file that references:

- library integrated information model 25, defined by the **ISO13584_25_liim_schema** EXPRESS schema and some of the conformance classes associated with constraints defined by **ISO_13584_cci_conformance_schema** and
- some view exchange protocols V1, V2,..., Vn, associated with constraints defined by schemas S_V1, ..., S_Vn, is implicitly defined as follows:

```

SCHEMA ISO13584_25_library_implicit_schema ;
  USE FROM ISO13584_25_liim_schema;
  REFERENCE FROM ISO13584_25_conformance_schema;
  REFERENCE FROM S_V1;
  REFERENCE FROM S_V2;
  ...
  REFERENCE FROM S_Vn;
END_SCHEMA; -- ISO13584_25_library_implicit_schema

```

Moreover, according to its conformance class, only instances of the entities defined in the corresponding conformance class schema shall exist in the library delivery file

NOTE In this part of ISO 13584, the constraints on conformance classes 1 to 4 are defined by the ISO13584_25_conformance_schema and no constraint exists on conformance class 0.

Whatever the conformance class a library delivery file claims conformance to, the name of the referenced schema shall be ISO13584_25_library_implicit_schema and no ISO13584_25_cci_schema (with i =0, 1, 2, 3 or 4).

B.2 Conformance class specification table

Table B.1 specifies the values of **library_liim_id.name** and **library_liim_id.application** that are allowed for use in a **library_liim_id** to reference library integrated information model 25 in either of its conformance classes.

Table B. 1 — ISO 13584 LIIM 25 conformance class specification

Conformance Class	library_liim_id.name mandatory value	library_liim_id.application mandatory value
0	'ISO_13584_25'	'0'
1	'ISO_13584_25'	'1'
2	'ISO_13584_25'	'2'
3	'ISO_13584_25'	'3'
4	'ISO_13584_25'	'4'

B.3 Standard data for conformance class 0

None.

B.4 Standard data for conformance class 1 to 4 (all the conformance classes but conformance class 0)

B.4.1. Constraints on a library delivery file conform to the library integrated model LIIM 25.

The **library_iim_id** instance values allowed for use in a library delivery file conform to the library integrated model LIIM 25 defined in this part of ISO 13584 in any conformance class 1 to 4 shall obey the constraints defined in the following EXPRESS schema.

EXPRESS specification:

```

*)
SCHEMA ISO13584_25_conformance_schema;

USE FROM ISO13584_IEC61360_dictionary_schema (
    item_names );

USE FROM ISO13584_IEC61360_language_resource_schema(
    translated_label);

USE FROM person_organization_schema(
    organization );

USE FROM support_resource_schema(
    label);

USE FROM ISO13584_extended_dictionary_schema(
    data_exchange_specification_id,
    library_iim_id);

USE FROM ISO13584_external_file_schema(
    http_protocol,
    standard_data_protocol,
    external_file_protocol);

(*

```

NOTE The schema used above can be found in the following document:
ISO13584_IEC61360_dictionary_schema IEC 61360-2:1998
 (which is duplicated for convenience in Informative Annex D of ISO 13584-42: 1998)
person_organization_schema ISO 10303-41:1994.
ISO13584_extended_dictionary_schema This part of ISO 13584.
ISO13584_external_file_schema This part of ISO 13584.

B.4.1.1 Allowed_reference_to_LIIM_25_rule rule

The **allowed_reference_to_LIIM_25_rule** rule defines a formal constraint and an informal constraint on **library_iim_ids** to be allowed for use to reference conformance class 1 to 4 of library integrated model LIIM 25 defined in this part of ISO 13584. A **library_iim_id** is allowed for use to reference conformance class 1 to 4 of library integrated model LIIM 25 if the following conditions hold:

- the **name** attribute of the **library_iim_id** that reference library integrated model LIIM 25 shall be equal to 'ISO_13584_25', and
- the **status** attribute of the **library_iim_id** shall be equal to either, 'CD' or 'DIS' or 'FDIS' or 'IS', and

- the **application** attribute of the **library_iim_id** shall have '1', '2', '3' or '4' and
- the **external_file_protocols** referenced by the **external_file_protocols** attribute of the **library_iim_id** shall fulfill the constraints required by the **compliant_external_file_protocol_25** function.

Moreover, a **library_iim_id** is allowed for use to reference conformance class 1 to 4 of library integrated model LIIM 25 if one of the two following conditions hold concerning the **http_files** that may be referenced directly or indirectly from the **library_iim_id**,

- either each referenced **http_file** it is associated with a **mime** attribute and an **exchange_format** attribute corresponding to MIME type and subtype that correspond to a specification that is publicly available, or
- it is associated with a **mime** attribute and an **exchange_format** attribute corresponding to MIME type and subtype that correspond to a specification that is associated with public domain Internet-available readers.

Reference to **http_files** corresponding to other MIME types and subtypes may only be done in extended conformance classes. This is documented as an informal proposition **IP1** in **allowed_reference_to_LIIM_25_rule** rule.

EXPRESS specification:

```

*)
RULE allowed_reference_to_LIIM_25_rule FOR (
    library_iim_id);
WHERE
    WR1: QUERY( liim_id <* library_iim_id |
        ((liim_id\data_exchange_specification_id.status = 'WD')
        OR
        (liim_id\data_exchange_specification_id.status = 'CD')
        OR
        (liim_id\data_exchange_specification_id.status = 'DIS')
        OR
        (liim_id\data_exchange_specification_id.status = 'FDIS')
        OR
        (liim_id\data_exchange_specification_id.status = 'IS')
    )
    AND
        (liim_id\data_exchange_specification_id.name
        = 'ISO_13584_25')
    AND
        is_correct_liim_25_application_value(liim_id)
    AND
        (QUERY( efp <* liim_id\data_exchange_specification_id
            .external_file_protocols
            | NOT(compliant_external_file_protocol_25([efp]))
        ) = [])
        =
        QUERY( liim_id <* library_iim_id |
            (liim_id\data_exchange_specification_id.name =
            'ISO_13584_25'));
END_RULE; -- allowed_reference_to_LIIM_25_rule
(*

```

Formal proposition:

WR1: when referencing library integrated model LIIM 25 defined in this part of ISO 13584, the **library_iim_id.name** shall have 'ISO_13584_25' as its value, **library_iim_id.status** shall be equal to either 'WD', 'CD' or 'DIS' or 'FDIS' or 'IS', the **library_iim_id.application** shall have '1', '2', '3' or '4' as its value, and the **library_iim_id.external_file_protocols** shall fulfill the constraints specifications required by the **compliant_external_file_protocol_25** function defined below.

Informal proposition:

IP1: when it references library integrated model LIIM 25 defined in this part of ISO 13584 in one of the conformance class 1, 2, 3, or 4, a **library_iim_id** may only reference, directly or indirectly, **http_files** characterized by MIME types and subtypes that either correspond to specifications that are publicly available, or to specifications that are associated with public domain Internet-available readers.

B.4.1.2 compliant_http_protocol_25 function

The **compliant_http_protocol_25** function checks whether an **external_file_protocol** may be referenced as the HTTP protocol by a **library_iim_id** that references library integrated model LIIM 25 in any of its conformance classes, or not. It returns TRUE if the given **external_file_protocol** is allowed for reference, otherwise, it returns FALSE. An **external_file_protocol** may be referenced as the HTTP protocol by a **library_iim_id** that reference library integrated model LIIM 25 in any of its conformance classes if the following conditions hold:

- the **external_file_protocol** shall be a **http_protocol**, and
- the **organisation** attribute of the **external_file_protocol** shall reference an **organization** of which the **id** attribute equals to 'IAB' and the **name** attribute equals to 'Internet Architecture Board', and
- the **protocol_name** attribute of the **external_file_protocol** shall equal to 'HTTP', and
- the **designation** attribute of the **external_file_protocol** shall reference an **item_names** for which the **preferred_name** attribute equals to 'Hypertext Transfer Protocol' and the **short_name** attribute equals to 'RFC' followed by four digits and possibly some other characters.

EXPRESS specification:

```

*)  

FUNCTION compliant_http_protocol_25(ef : external_file_protocol)  

      : BOOLEAN;  

LOCAL  

  ok: BOOLEAN := TRUE;  

END_LOCAL ;  

IF (('ISO13584_EXTERNAL_FILE_SCHEMA'  

  + '.HTTP_PROTOCOL' IN TYPEOF(ef)) AND  

  (ef.organisation.id = 'IAB') AND  

  (ef.organisation.name = 'Internet Architecture Board') AND  

  (ef.protocol_name = 'HTTP') AND  

  (ef.designation.preferred_name  

  = 'Hypertext Transfer Protocol') )  

THEN

```

```

IF 'ISO13584_IEC61360_LANGUAGE_RESOURCE_SCHEMA.TRANSLATED_LABEL'
    IN TYPEOF(ef.designation.short_name)
THEN
    REPEAT i:= 1 TO SIZEOF(ef.designation.short_name
                           \translated_label.labels);
        IF (ef.designation.short_name\translated_label.labels[i]
            LIKE 'RFC####&')
    THEN
        ok := ok AND TRUE;
    ELSE
        ok := OK AND FALSE;
    END_IF;
END_REPEAT ;
RETURN (OK) ;
ELSE
    IF ef.designation.short_name
        LIKE 'RFC####&'
    THEN
        RETURN (TRUE) ;
    ELSE
        RETURN (FALSE) ;
    END_IF;
END_IF ;
ELSE
    RETURN (FALSE) ;
END_IF;

END_FUNCTION; -- compliant_http_protocol_25
(*

```

B.4.1.3 compliant_8859_1_protocol_25 function

The **compliant_8859_1_protocol_25** function checks whether an **external_file_protocol** may be referenced as the ISO 8859-1 protocol by a **library_iim_id** that references library integrated model LIIM 25 in any of its conformance classes, or not. It returns TRUE if the given **external_file_protocol** is allowed for reference, otherwise, it returns FALSE. An **external_file_protocol** may be referenced as the ISO 8859-1 protocol by a **library_iim_id** that represents reference library integrated model LIIM 25 in any of its conformance classes, if the following conditions hold:

- the **external_file_protocol** shall be a **standard_data_protocol**, and
- the **organisation** attribute of the **external_file_protocol** shall reference an **organization** of which the **id** attribute equals to 'ISO' and the **name** attribute equals to 'International Organisation for Standardization', and
- the **protocol_name** attribute of the **external_file_protocol** shall equal to 'ISO_8859_1', and
- the **designation** attribute of the **external_file_protocol** shall reference an **item_names** for which the **preferred_name** attribute equals to 'Latin alphabet No 1' and the **short_name** attribute equals to 'ISO 8859-1'.

EXPRESS specification:

```

*)
FUNCTION compliant_8859_1_protocol_25(ef : external_file_protocol)
      :BOOLEAN;

IF (('ISO13584_EXTERNAL_FILE_SCHEMA'
+ '.STANDARD_DATA_PROTOCOL' IN TYPEOF(ef)) AND
(ef.organisation.id = 'ISO') AND
(ef.organisation.name
= 'International Organisation for Standardization') AND
(ef.protocol_name = 'ISO_8859_1') AND
(ef.designation.preferred_name
= 'Latin alphabet No 1') AND
(ef.designation.short_name = 'ISO 8859-1'))
THEN
    RETURN(TRUE);
ELSE
    RETURN(FALSE);
END_IF;
END_FUNCTION; -- compliant_8859_1_protocol_25
(*

```

B.4.1.4 compliant_external_file_protocol_25 function

The **compliant_external_file_protocol_25** function checks whether all the **external_file_protocols** of a set of **external_file_protocols** may be referenced as an library integrated model LIIM 25 by a **library_iim_id** that references library integrated model LIIM 25 in one of its conformance class 1 to 4, or not. It returns TRUE if all the **external_file_protocols** of a set of **external_file_protocols** are allowed for reference, otherwise, it returns FALSE.

An **external_file_protocol** may be referenced by a **library_iim_id** that represents conformance class 1 to 4 of library integrated model LIIM 25 if it may be referenced:

- either as the HTTP protocol, or
- as the ISO 8859-1 protocol.

NOTE In extended conformance classes of library integrated model LIIM 25, any other **external_file_protocol** may be referenced, subject to private agreement between the sender and the receiver.

EXPRESS specification:

```

*)
FUNCTION compliant_external_file_protocol_25(
      S : SET [0:?] OF external_file_protocol)
      : BOOLEAN;

REPEAT i := 1 TO SIZEOF(S) ;
  IF NOT (compliant_8859_1_protocol_25 (S[i])
          OR compliant_http_protocol_25 (S[i]))
  THEN
    RETURN(FALSE);
  END_IF;

```

```

END_REPEAT;

RETURN (TRUE);

END_FUNCTION; -- compliant_external_file_protocol_25
(*

```

B.4.1.5 is_correct_liim_25_application_value function

The **is_correct_liim_25_application_value** function checks that the **liim_id library_iim_id** is compatible with the conformance classes associated to the LIMM 25.

EXPRESS specification:

```

*)
FUNCTION is_correct_liim_25_application_value
    (liim_id : library_iim_id) : BOOLEAN;

IF EXISTS(liim_id\data_exchange_specification_id.application)
    AND
        ((liim_id\data_exchange_specification_id.application[1]='1')
            OR
            (liim_id\data_exchange_specification_id.application[1]='2')
            OR
            (liim_id\data_exchange_specification_id.application[1]='3')
            OR
            (liim_id\data_exchange_specification_id.application[1]='4'))
    AND
        ((liim_id\data_exchange_specification_id.application
            LIKE '#'))
THEN
    RETURN (TRUE);
ELSE
    RETURN (FALSE);
END_IF;
END_FUNCTION; -- is_correct_liim_25_application_value
(*

```

Annex C (normative)

Implementation method specific requirements for the library integrated information model 25

Conformance to the library integrated information model 25 shall be realised in one or more implementation methods. The implementation methods defines what types of exchange behaviour is required with respect to exchange protocols.

One implementation method is defined for the library delivery file: ISO 10303-21.

The implementation methods for the possible external files referenced from the library delivery file and whose **external_file_protocol** belong to the standard data of the library integrated information model 25 are defined by the standard referenced in this **external_file_protocol**, possibly further specified as part of the description of the library integrated information model standard data (see Annexe B).

For the exchange structure, the file format of the library delivery file shall be encoded according to the syntax and EXPRESS language mapping defined in ISO 10303-21 for the schema defined in annex A of this part of ISO 13584. The header of the exchange structure shall identify use of this part of ISO 13584 by the schema names 'ISO13584_25_library_implicit_schema'.

NOTE Identification of the library delivery file is done by separate agreement between the sender and the receiver and is outside the scope of this part of ISO 13584.

Annex D (informative) EXPRESS-G diagrams

Figure D.1 through D.2 correspond to the EXPRESS schemas given in chapters 6 and 7. The diagrams use the EXPRESS-G graphical notation for the EXPRESS language. EXPRESS-G is defined in annex A of ISO 10303-11.

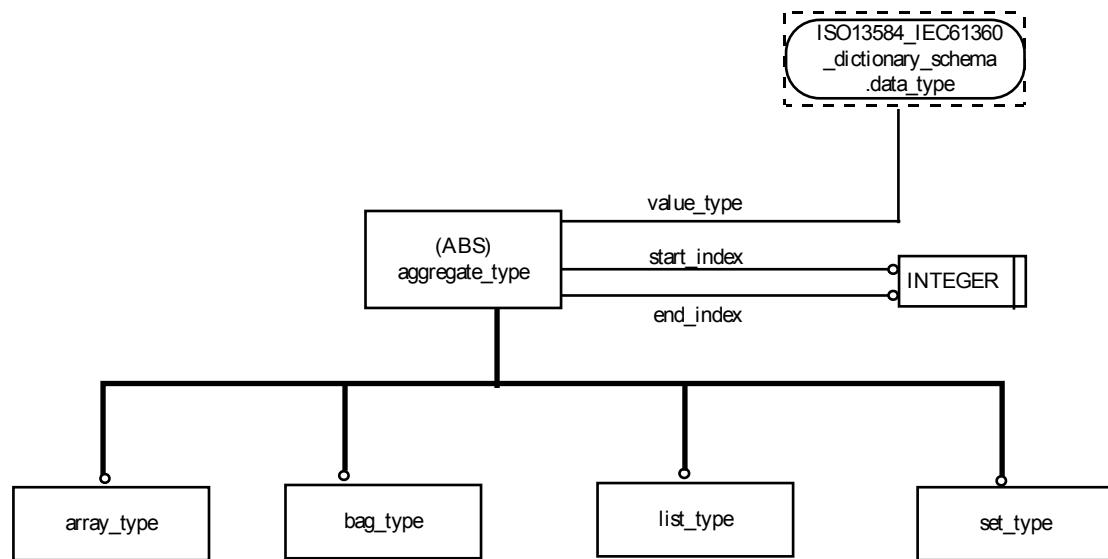


Figure D. 1 — ISO13584_IEC61360_dictionary_aggregate_extension_schema diagram 1 of 1

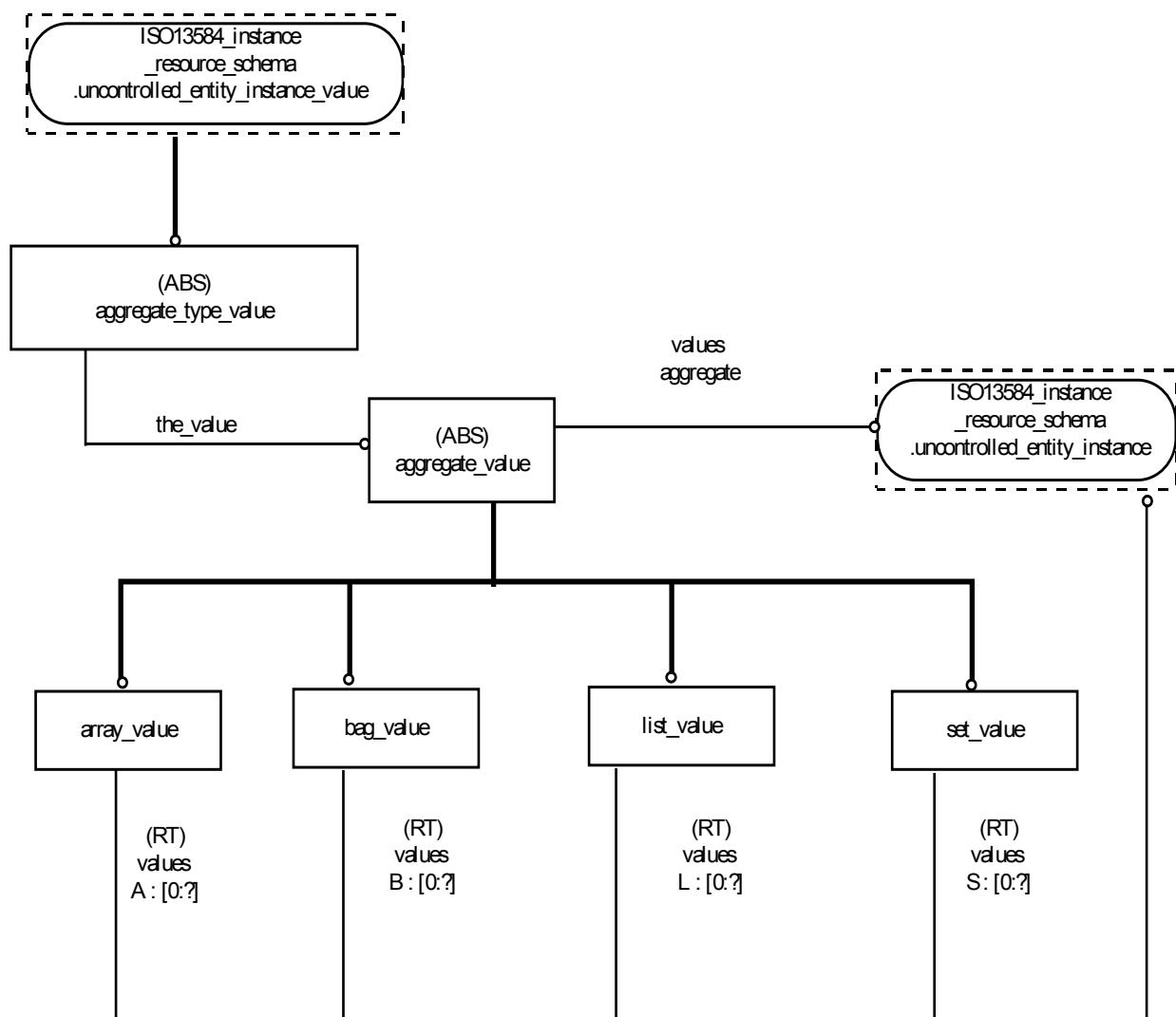


Figure D. 2 — ISO13584_dictionary_aggregate_value_schema diagram 1 of 1

Annex E (informative)
Commented example of library integrated information model 25
physical files.
Exchange of explicit general models

This annex presents, on an example, an overview of the different resources involved in the description of a parts library using an explicit modeling according to this part of ISO 13584.

E.1 Capturing a parts family in ISO 13584

Figure E.1 illustrates the part family intended to be described. It is a family of washers, denoted PAW, that is sold by a bearing supplier and that is used as a bearing in some mechanical contexts.

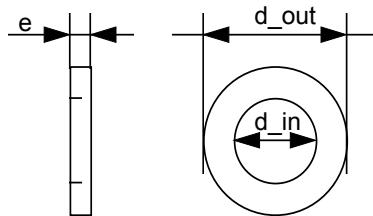


Figure E. 1 — PAW family description

Such a part family may be described at two levels of abstraction:

The dictionary level (**ISO13584_IEC61360_dictionary_schema** and **ISO13584_extended_dictionary_schema**) permits the description of the concepts of a part family i.e, the supplier(s), the class(es), the property(ies), It defines what are the meaning and the value types of the properties, in which classes the properties are visible, by which supplier the classes are specified... . Such a data **dictionary_element** may be done both for the general model description (what is this family of parts) and for the functional model description (what kind of representations may be defined for this family of parts).

The library level (**ISO13584_library_content_schema**) permits the definition of the values of the properties to explicitly define the allowed instances of the parts family. This level is for instance useful when describing an extension of a library.

These two levels of abstraction have to be represented in two different ways.

The following Figure (Figure P.2) presents an example of instance of a family that is only defined at the dictionary level. The permitted instances are those where the values of the properties describing the part belong to the type defined in the data dictionary.

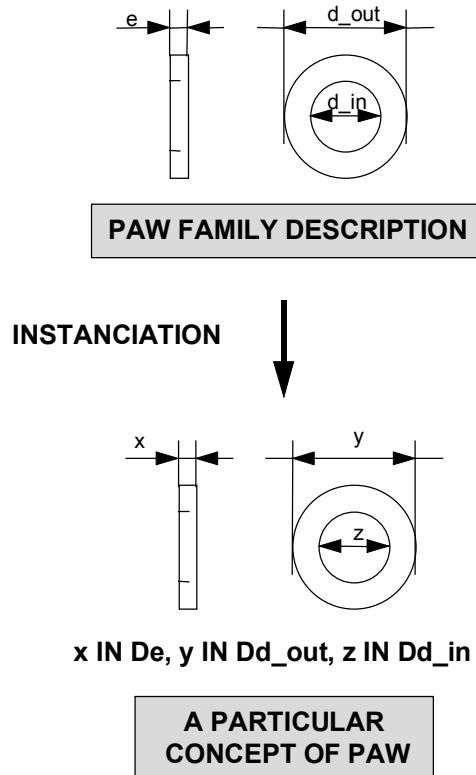
**Figure E. 2 — Instance of a dictionary description**

Figure P.3 describes and presents an example of a family that is associated with a two-fold description: **dictionary_element** and library specification. The allowed instances are those where the values of the properties describing the part belong to the (explicit) list of authorized tuples of values defined in the library specification of a part.

d_in	e	d_out
10	1	15
11	1	16.5
13	2	19.5
17	3	25.5
19	4	28.5

Figure E. 3 — Explicit description of a dictionary description

The example presented in this annex involves this kind of two-fold description because this part family is defined originally in a paper catalogue. Therefore only a well-defined set of instances are really provided for the bearing.

E.2 Description of the PAW parts family

This description is two-fold. First the concept of PAW and of its properties are defined. Second, the allowed instances are precisely (but implicitly) defined.

E.2.1 Dictionary description: the BSU mechanism

The description of a data dictionary according to the **ISO13584_IEC61360_dictionary_schema** schema requires the specification of what are the identifiers of the different concepts (called basic semantic unit: BSU) involved in the parts family definition. These identifiers define unambiguously and universally each concept within an ISO 13584-compliant data dictionary.

The following example (Figure E.4) outlines the resources used for the specification of these identifiers:

```

/*BSU for supplier */
/* The code of the supplier must be defined according to ISO13584-26: Supplier identification
   Here, the code doesn't follow the ISO 13584-26 requirements, because the supplier code is not
   known at the moment*/
#20 = SUPPLIER_BSU ('INA', *);

/* BSU for component_class */
/* The class BSUs defines the identification of the various classes, and who is the supplier that is
   responsible of the class definitions */
#50 = CLASS_BSU ('BEARING', '001', #20);
#60 = CLASS_BSU ('PAW', '001', #20);

/* BSU for properties */
/* The property BSU defines the identifications of the properties and the class where these
   properties are visible */
#90 = PROPERTY_BSU ('d_in', '001', #50);
#100 = PROPERTY_BSU ('d_out', '001', #50);
#110 = PROPERTY_BSU ('e', '001', #60);

```

Figure E. 4 — Identifiers of the concepts involved in the PAW family

E.2.2 Dictionary description: the dictionary element definition

A BSU only identifies a concept. A **dictionary_element** provides a computer-sensible and human-readable definition of the concept. This relationship between these two levels is presented in Figure E.5.

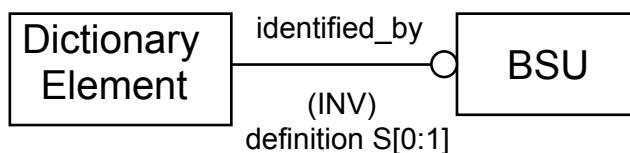


Figure E. 5 — The BSU / Dictionary element relationship

The following Figure outlines the main structure of the **dictionary_elements** corresponding to the previous basic semantic units identifiers.

```

/* Dictionary properties description */
/* supplier description */
#21 = SUPPLIER_ELEMENT (#20, $, '001', #22, #23);
#22 = ORGANIZATION ($, 'INA', '');

```

```

#23 = ADDRESS ($, $, $, $, $, $, $, '$GERMANY', $, $, $, $);

/* d_in */
#91 = NON_DEPENDENT_P_DET (#90, $, '001',
    #92, /* Item names associated to the non dependent P_DET */
    TEXT('inner diameter'), $, $, $, $, (), $, 'TO3',
    #93, /* the specific data type of the property (not represented:measure
    in mm) */
    $);
#92 = ITEM_NAMES (LABEL('inner diameter'), (), LABEL(''), $, $);
#93 = REAL_MEASURE_TYPE ('NR2..3.3', #94);
#94 = DIC_UNIT (#95, $);
#95 = SI_UNIT (*, .MILLI., .METRE.);

/* d_out */
#101 = NON_DEPENDENT_P_DET (#100, $, '001', #102, TEXT('outer diameter'),
$, $, $, $, $, $, 'TO3', #93, $);

/* e */
#111 = NON_DEPENDENT_P_DET (#110, $, '001', #112, TEXT('thickness'), $, $,
$, $, $, $, 'TO3', #93, $);

/* Dictionary class description */
/* Part class */
#71 = COMPONENT_CLASS (#50, /* reference to its BSU */
    $, '001', #72, /* item_names */
    TEXT('Class associated to the generic bearing family'),
    /* Definition */
    $$,$,$,(#90,#100), /* the list of the properties
        that may be used to describe an instance of this class
        (applicability of the properties) */
    (), $, (), (), $);
#72 = ITEM_NAMES (LABEL('Generic bearing family'), (), LABEL('Bearing
family'), $,
$);

/* PAW class */
#81 = COMPONENT_CLASS (#60, $, '001', #82, TEXT('Class associated to the
PAW
part family'), $, $, $, #50, (#110), (), $, (), (), $);

```

Figure E. 6 — Dictionary_element of the concepts involved in the PAW family

E.2.3 Library specification: description of the class extension

The extension of a class is given by the set of instances of a class. Each instance of a class contains a set of property values that correspond to the values of the properties belonging to the part family described by this instance. Figure E.7 Describes the extension of a class by an explicit modeling.

```

/* Dictionary extension */
/* Extension of a class */
#8000= EXPLICIT_ITEM_CLASS_EXTENSION(#60, /*Reference to the BSU */
    (),(),(),'001','001',(),(#90),
    (#8100,#8200,#8300,#8400,#8500),/* the extension of class
    given by a list of class instances.

```

```
T., $, $, (), $, (), () ;
```

Figure E. 7 — Dictionary_element of the concepts involved in the PAW family

The class instance is described by the list of its property values. Figure E.8 gives the description, by extension of one instance corresponding to the part family for which **d_in** equals 10, **e** equals 1 and **d_out** equals 15.

```
/* Extension of a library component */
#8100=LIB_COMPONENT_INSTANCE(#60, /* the BSU associated to the class which
                                     the current instance is an instance of */
                                (#8101, #8102, #8103), /* the property values */
                                (), $, $, $, $, $);

/* Property values of the extension of a class */
#8101=PROPERTY_VALUE(REAL_VALUE(10), #90);
#8102=PROPERTY_VALUE(REAL_VALUE(1), #100);
#8103=PROPERTY_VALUE(REAL_VALUE(15), #110);
```

Figure E. 7 — Dictionary_element of the concepts involved in the PAW family

E.3 A complete physical file for explicit general models.

In this clause, the complete example of a physical file is provided. The following physical file, compliant with LIBRARY INTEGRATED INFORMATION MODEL 25 defines the PAW family.

```
ISO-10303-21;
HEADER;

FILE_DESCRIPTION (('THIS IS AN EXAMPLE OF AN EXPLICIT GENERAL MODEL'),
'2');
FILE_NAME('P25_gm_explicit.p21',
          '2000-11-28T17:38:14',
          (''),
          ('LISI/ENSMA'),
          'ECCO RUNTIME SYSTEM BUILT-IN PREPROCESSOR V2.3beta1',
          'ECCO RUNTIME SYSTEM V2.3beta1',
          '');
FILE_SCHEMA (('ISO_13584_CC3_CONFORMANCE_SCHEMA'));
ENDSEC;

DATA;

/* Global library description */
#2 = LIBRARY_IN_STANDARD_FORMAT (#20, #11, (), (), (#20), (#50, #60),
(), #3, $, $, ());
#3 = ITEM_NAMES (LABEL('Explicit general model example'), (), LABEL(''), $,
$);
#10 = GLOBAL_LANGUAGE_ASSIGNMENT ('en');
#11 = LIBRARY_IIM_IDENTIFICATION ('WD', 'ISO13584_25', 2001, '3', $, ());

/* DICTIONARY DESCRIPTION */
/*BSU for supplier */

#20 = SUPPLIER_BSU ('INA', *);

/* BSU for component_class */
#50 = CLASS_BSU ('BEARING', '001', #20);
#60 = CLASS_BSU ('PAW', '001', #20);
```

```

/* BSU for properties */
#90 = PROPERTY_BSU ('d_in', '001', #50);
#100 = PROPERTY_BSU ('d_out', '001', #50);
#110 = PROPERTY_BSU ('e', '001', #60);

/* Dictionary properties description */
/* supplier description */
#21 = SUPPLIER_ELEMENT (#20, $, '001', #22, #23);
#22 = ORGANIZATION ($, 'INA', '');
#23 = ADDRESS ($, $, $, $, $, $, $, 'GERMANY', $, $, $, $);

/* d_in */
#91 = NON_DEPENDENT_P_DET (#90, $, '001', #92, TEXT('inner diameter'), $,
$, $, $, $, 'TO3', #93, $);
#92 = ITEM_NAMES (LABEL('inner diameter'), (), LABEL(''), $, $);
#93 = REAL_MEASURE_TYPE ('NR2..3.3', #94);
#94 = DIC_UNIT (#95, $);
#95 = SI_UNIT (*, .MILLI., .METRE.);

/* d_out */
#101 = NON_DEPENDENT_P_DET (#100, $, '001', #102, TEXT('outer diameter'),
$, $, $, $, 'TO3', #93, $);
#102 = ITEM_NAMES (LABEL('outer diameter'), (), LABEL(''), $, $);
#103 = REAL_MEASURE_TYPE ('NR2..3.3', #104);
#104 = DIC_UNIT (#105, $);
#105 = SI_UNIT (*, .MILLI., .METRE.);

/* e */
#111 = NON_DEPENDENT_P_DET (#110, $, '001', #112, TEXT('thickness'), $, $,
$, $, $, 'TO3', #93, $);
#112 = ITEM_NAMES (LABEL('thickness'), (), LABEL(''), $, $);
#113 = REAL_MEASURE_TYPE ('NR2..3.3', #114);
#114 = DIC_UNIT (#115, $);
#115 = SI_UNIT (*, .MILLI., .METRE.);

/* Dictionary class description */
/* Part class */
#71 = COMPONENT_CLASS (#50, $, '001', #72, TEXT('Class associated to the
generic bearing family'), $, $, $, $, (#90, #100), (), $, (), $);
#72 = ITEM_NAMES (LABEL('Generic bearing family'), (), LABEL('Bearing
family'), $,
$);

/* PAW class */
#81 = COMPONENT_CLASS (#60, $, '001', #82, TEXT('Class associated to the
PAW
part family'), $, $, $, #50, (#110), (), $, (), $);
#82 = ITEM_NAMES (LABEL('PAW family'), (), LABEL('PAW'), $, $);

/* Dictionary extension */
/* Extension of a class */
#8000= EXPLICIT_ITEM_CLASS_EXTENSION(#60,(),(),(),'001','001',(),(#90),(
#8100,#8200,#8300,#8400,#8500),.T.,$,,$,(),$,(),());

/* Extension of a library component */

```

```

#8100=LIB_COMPONENT_INSTANCE(#60, (#8101, #8102, #8103), (), $, $, $, $,
$);
/* Property values of the extension of a class */
#8101=PROPERTY_VALUE(REAL_VALUE(10), #90);
#8102=PROPERTY_VALUE(REAL_VALUE(1), #100);
#8103=PROPERTY_VALUE(REAL_VALUE(15), #110);

/* Extension of a library component */
#8200=LIB_COMPONENT_INSTANCE(#60, (#8201, #8202, #8203), (), $, $, $, $,
$);
/* Property values of the extension of a class */
#8201=PROPERTY_VALUE(REAL_VALUE(11), #90);
#8202=PROPERTY_VALUE(REAL_VALUE(1), #100);
#8203=PROPERTY_VALUE(REAL_VALUE(16.5), #110);

/* Extension of a library component */
#8300=LIB_COMPONENT_INSTANCE(#60, (#8301, #8302, #8303), (), $, $, $, $,
$);
/* Property values of the extension of a class */
#8301=PROPERTY_VALUE(REAL_VALUE(13), #90);
#8302=PROPERTY_VALUE(REAL_VALUE(2), #100);
#8303=PROPERTY_VALUE(REAL_VALUE(19.5), #110);

/* Extension of a library component */
#8400=LIB_COMPONENT_INSTANCE(#60, (#8401, #8402, #8403), (), $, $, $, $,
$);
/* Property values of the extension of a class */
#8401=PROPERTY_VALUE(REAL_VALUE(17), #90);
#8402=PROPERTY_VALUE(REAL_VALUE(3), #100);
#8403=PROPERTY_VALUE(REAL_VALUE(25.5), #110);

/* Extension of a library component */
#8500=LIB_COMPONENT_INSTANCE(#60, (#8501, #8502, #8503), (), $, $, $, $,
$);

/* Property values of the extension of a class */
#8501=PROPERTY_VALUE(REAL_VALUE(19), #90);
#8502=PROPERTY_VALUE(REAL_VALUE(4), #100);
#8503=PROPERTY_VALUE(REAL_VALUE(28.5), #110);

ENDSEC;
END-ISO-10303-21;

```

Annex F (informative)
Commented example of library integrated information model 25
physical files.
Exchange of explicit functional models compliant with ISO 13584-
101

This annex presents, on an example, an overview of the different resources involved in the description of a parts library using an explicit modeling according to this part of ISO 13584. It describes a physical file example allowing the exchange of explicit functional models compliant with ISO 13584 – 101 view exchange protocol.

F.1 Description of the PAW parts family and its geometry

The description of the Paw family is identical to the one described and commented in the previous annex E except that supplier elements have been introduced in order to describe the view and the geometry suppliers.

```

/* DICTIONARY DESCRIPTION */

/* The two supplier_BSUs LISI/ENSMA and ISO identify respectively the suppliers of the functional
model and of the functional view. */

/*BSU for supplier */
#20=SUPPLIER_BSU('INA', *); /* INA code unknown */
    /* Supplier of the part family */
#30=SUPPLIER_BSU('9/19860073600021', *);
    /* LISI/ENSMA code in the coding scheme ICD=0009 : SIRET number */
    /* supplier of the functional model class */

#40=SUPPLIER_BSU('0112/1///13584_101_1', *);
    /* Identification of ISO 13584-101 according to ISO 13584-26 */
    /* supplier of the functional view */

/* BSU for component_class */
/*The two class_BSUs PAW_Gemetry and basic_geometry identify respectively the classes
corresponding to the functional model and of the functional view. */

#50=CLASS_BSU('Bearing', '001', #20);
#60=CLASS_BSU('PAW', '001', #20);
#130=CLASS_BSU('PAW_Geometry', '001', #30);
#140=CLASS_BSU('basic_geometry', '001', #40);

/* BSU for properties */
/* The follwing property_BSU describe the properties of the part family PAW as defined in annex E.*/

#90=PROPERTY_BSU('d_in', '001', #50);
#100=PROPERTY_BSU('d_out', '001', #50);
#110=PROPERTY_BSU('e', '001', #50);

/* The definition of the geometry for a given part, and particularly for the part family PAW requires the
definition of representation properties. These properties are defined as for part family properties
```

through the BSU mechanism. However, this BSU identifies a **representation_P_DET** element.*/

```
#150=PROPERTY_BSU('geometry_level', '001', #140);
#160=PROPERTY_BSU('detail_level', '001', #140);
#170=PROPERTY_BSU('side', '001', #140);
#180=PROPERTY_BSU('prg', '001', #130);
#200=PROPERTY_BSU('variant', '001', #140);
#210=PROPERTY_BSU('unreg_variant', '001', #140);

/* supplier description */
/* The following supplier_element describes LISI/ENSMA as supplier. It will be used as supplier of a functional model class. */

#31=SUPPLIER_ELEMENT(#30, $, '001', #32, #33);
#32=ORGANIZATION('LISI/ENSMA', 'LISI/ENSMA', '');
#33=ADDRESS($, $, $, $, $, $, $, 'FRANCE', $, $, $, $);

/* Dictionary properties description */
/* prg */

/*Data type elements associated to representation properties are representation_P_DETs. As example; the following data element describes the variable allowing to refer programs (prg). Its values will be described below. */

#91=REPRESENTATION_P_DET (#180,
    $, '001', #92,
    TEXT('variable used to reference geometry programs'), $, $, $, $,
    (), $, 'A58', #93, $);
#92=ITEM NAMES (LABEL('related program'), (), LABEL(''), $, $);
#93=PROGRAM_REFERENCE_TYPE (
    'ISO13584_EXTERNAL_FILE_SCHEMA.PROGRAM_REFERENCE');


```

Figure F.1 — The Identifiers of the concepts involved in the Paw family and its geometry representation

F.2. Description of geometric representations for the PAW parts family

We assume now that some library data supplier wants to provide a geometric representation for all the instances of the PAW family that are described in an explicit manner (by extension). This requires the description of a functional model class.

A functional model class is intended to represent different perspectives of the different parts described in the general model class. A functional model class has to be described like a general model class, i.e., through a class definition and through a dictionary extension (library specification).

A **functional_model_class** describes a particular view (is-view-of relationship) of a given parts family (described as a general model class), according to the point of view specified by a **functional_view_class**.

In the example, it will be defined a functional model class representing some kind of geometry (specified by) a **functional_view_class** for the PAW parts family.

A **fm_class_view_of** is a functional model class that refers to a well defined general model class (here the class that models the PAW family) and that provides a particular kind of representation (specified by a **functional_view_class**) for this general model class.

A functional model class is not required to provide representation for all the values of the functional view class view control variables. The range of supported values is specified by

view_control_variable_range as shown in Figure F.2.

```
/* v_c_v range */
#155=VIEW_CONTROL_VARIABLE_RANGE(#150, 1, 1);
#165=VIEW_CONTROL_VARIABLE_RANGE(#160, 2, 2);
#175=VIEW_CONTROL_VARIABLE_RANGE(#170, 1, 6);
#205=VIEW_CONTROL_VARIABLE_RANGE (#200, 1, 1);
#215=VIEW_CONTROL_VARIABLE_RANGE (#210, 0, 0);
```

Figure F.2 — View control variables range definition

In our example, the functional model class only provides 2D views (range 1..1, for #150 that is 'geometry_level'), with standard representation (range 2..2, for #160 that is 'detail-level') for all the sides from 'front' to 'bottom' (range 1..6, for # 170 that is 'side').

Moreover, to be able to create the geometry, the **fm_class_view_of** needs to import some properties of the PAW family. The **dictionary_element** of the **fm_class_view_of** is presented in the Figure F.3.

```
/* Dictionary class description */
/* Functional model class view_of definition*/
/* The following instance describes the is_view_of relationship through a fm_class_view_of class
supplied by LISI/ENSMIA.

#71=FM_CLASS_VIEW_OF(#130, /* reference to BSU */
$ , '001', #72, /* item names */
TEXT('Explicit functional model class describing the 2d
standard geometry of PAW'), $, $, $, $,
(#180), /* BSU of the 'prg'and of the required_side properties */
(), *, *, *, *, *,
#140, the created view (reference to the BSU of the
functional_view_class */
(#155, #165, #175, #205, #215), /* the vcv ranges */
(#150, #160, #170, #200, #210), /* vcv's imported from the functional
view class */
(), (), (), #60, /* is_view_of relationship. Reference to the
functional model */
(#90, #100, #110), /* imported properties from the general model */
(), (), ());
#72=ITEM_NAMES(LABEL('Functional model class of PAW'), (), LABEL('fm class
of PAW'), $, $);
```

Figure F.3. — Specification of the view created by a functional model class

F.3. Library specification of the functional model class

A **explicit_functional_model_class_extension** is the **content_item** that constitutes the library specification a **functional_model_class** and **fm_class_view_of** subtype). It gives the set of the properties that need to be valued in the context of an instance of such a class

In the case of an explicit representation, the instances of a functional model are explitly enumerated. In the following **explict_functional_model_extension** instance, the instances of a functional model are described by **lib_F_model_instances** in the range 3000 to 3450.

```
/* LIBRARY DESCRIPTION */
/* Description of the extension of a functional model
```

```

class */
/* It references all the library functional model
instances described themselves by extesion */
/* In the case of an explicit representation, the instances of a functional model are explitly enumerated.
In the following explict_functional_model_extension instance, the instances of a functional model
are described by lib_F_model_instances in the range 3000 to 3450.*/
#1300=EXPLICIT_FUNCTIONAL_MODEL_CLASS_EXTENSION(#130,
    (#2501, #2502, #2503, #2504, #2505, #2506), /* the set of program
                                                 references. The programs
                                                 which allow to display
                                                 geometry.*/
    (#7), (#12), '001', '001', (),
    (#90, #170), /* properties needed to display the geometry */
    (#3000, #3010, #3020, #3030, #3040, #3050,
     #3100, #3110, #3120, #3130, #3140, #3150,
     #3200, #3210, #3220, #3230, #3240, #3250,
     #3300, #3310, #3320, #3330, #3340, #3350,
     #3400, #3410, #3420, #3430, #3440, #3450), /* The extension of all the
                                                 instances of a
                                                 functional model. They are
                                                 given by the
                                                 lib_F_model_instances
.T., $, (#90), (), $, $, (), $);

```

Figure F.4. — Description by extension of the instances of a functional a functional model

```

/* Reference to programs which display geometry */
/* According to the previous instance, references to programs that describe geometry representation
are performed. These program_references refer themselves to external files PAW_p1.for ..
PAW_p6.for containing geometry programs written in the FORTRAN language according to view
exchange protocol ISO 13584-101.*/
#2501=PROGRAM_REFERENCE(#7, #2601,
    'Add1_PAW', 'PAW_p1',
    (#90, #100, #110),
    $, $);
#2502=PROGRAM_REFERENCE(#7, #2602, 'Add2_PAW', 'PAW_p2', (#90, #100, #110),
$,$);
#2503=PROGRAM_REFERENCE(#7, #2603, 'Add3_PAW', 'PAW_p3', (#90, #100, #110),
$,$);
#2504=PROGRAM_REFERENCE(#7, #2604, 'Add4_PAW', 'PAW_p4', (#90, #100, #110),
$,$);
#2505=PROGRAM_REFERENCE(#7, #2605, 'Add5_PAW', 'PAW_p5', (#90, #100, #110),
$,$);
#2506=PROGRAM_REFERENCE(#7, #2606, 'Add6_PAW', 'PAW_p6', (#90, #100, #110),
$,$);

#2601=NOT_TRANSLATABLE_EXTERNAL_CONTENT((#2701));
#2602=NOT_TRANSLATABLE_EXTERNAL_CONTENT((#2702));
#2603=NOT_TRANSLATABLE_EXTERNAL_CONTENT((#2703));
#2604=NOT_TRANSLATABLE_EXTERNAL_CONTENT((#2704));
#2605=NOT_TRANSLATABLE_EXTERNAL_CONTENT((#2705));
#2606=NOT_TRANSLATABLE_EXTERNAL_CONTENT((#2706));
#2701=LANGUAGE_SPECIFIC_CONTENT((#2801), #2801, $);
#2702=LANGUAGE_SPECIFIC_CONTENT((#2802), #2802, $);
#2703=LANGUAGE_SPECIFIC_CONTENT((#2803), #2803, $);
#2704=LANGUAGE_SPECIFIC_CONTENT((#2804), #2804, $);
#2705=LANGUAGE_SPECIFIC_CONTENT((#2805), #2805, $);
#2706=LANGUAGE_SPECIFIC_CONTENT((#2806), #2806, $);

```

```

/* Description of the source files for geometry */
#2801=EXTERNAL_FILE_UNIT('PAW_p1.for', '7bit');
#2802=EXTERNAL_FILE_UNIT('PAW_p2.for', '7bit');
#2803=EXTERNAL_FILE_UNIT('PAW_p3.for', '7bit');
#2804=EXTERNAL_FILE_UNIT('PAW_p4.for', '7bit');
#2805=EXTERNAL_FILE_UNIT('PAW_p5.for', '7bit');
#2806=EXTERNAL_FILE_UNIT('PAW_p6.for', '7bit');

```

Figure F.5. — References to FORTRAN programs that display geometry.

Instances of functional models described by extension, describe themselves their property values by extension. Figure F.6 shows one instance of functional model which describe by extension the values of its properties.

```

/* Description of library functional model instance
by extension */

#3000=LIB_F_MODEL_INSTANCE(#130, (#3001, #3002, #3003, #3004, #3005, #3006,
#3007), ());

/* property values that describe the previous library
functional model extension */

#3001=PROPERTY_VALUE(REAL_VALUE(10), #90);
#3002=PROPERTY_VALUE(1, #170);
#3003=PROPERTY_VALUE(1, #150);
#3004=PROPERTY_VALUE(2, #160);
#3005=PROPERTY_VALUE(1, #200);
#3006=PROPERTY_VALUE(0, #210);
#3007=PROPERTY_VALUE(#2501, #180);

```

Figure F.6. — The BSU / Dictionary element relationship

F.4 A complete physical file for explicit functional models compliant with ISO 13584-101.

In this clause, the complete example of a physical file is provided. The following physical file, compliant with LIBRARY INTEGRATED INFORMATION MODEL 25 defines the PAW family and its geometry according to the view exchange protocol defined in ISO 13584 Part 101 which references the **programs** that allows to display geometry.

```

ISO-10303-21;
HEADER;
FILE_DESCRIPTION(('PLIB EXPLICIT FUNCTIONAL MODEL EXAMPLE 1'), '1');
FILE_NAME('P25_fm_explicit.p21',
          '2000-11-28T18:38:14',
          (''),
          ('LISI/ENSMA'),
          'ECCO RUNTIME SYSTEM BUILT-IN PREPROCESSOR V2.3beta1',
          'ECCO RUNTIME SYSTEM V2.3beta1',
          '');
FILE_SCHEMA(('ISO_13584_CC3_CONFORMANCE_SCHEMA'));
ENDSEC;

```

DATA;

```

/* Global library description */
#2=LIBRARY_IN_STANDARD_FORMAT(#30, #11, (#7), (#12), (#20, #30,
#40), (#50, #60, #140, #130), (), #3, $, $, ());
#3=ITEM_NAMES(LABEL('Explicit functional model: Geometry'), (),
LABEL('Geometry'), $,
$);
#6=ORGANIZATION('LISI/ENSMA', 'LISI/ENSMA', '');
#7=STANDARD_SIMPLE_PROGRAM_PROTOCOL(#6, $, 'ISO_IS_13584_31',
'001', $, #8, $, 'FORTRAN', .SOURCE., $, $, $);
#8=ITEM_NAMES(LABEL('Geometric prog. interface'), (),
LABEL('ISO_IS_13584_31'), $, $);
#11=LIBRARY_IIM_IDENTIFICATION('WD', 'ISO13584_25', 2001, '3', $, ());
#12=VIEW_EXCHANGE_PROTOCOL_IDENTIFICATION('DIS', 'ISO13584_101', 1998,
'2D', $,
(#7), $);
#10=GLOBAL_LANGUAGE_ASSIGNMENT('en');

/* DICTIONARY DESCRIPTION */
/*BSU for supplier */
#20=SUPPLIER_BSU('INA', *); /* INA code unknown */
#30=SUPPLIER_BSU('9/19860073600021', *);
/* LISI/ENSMA code in the coding scheme ICD=0009 : SIRET number */
#40=SUPPLIER_BSU('0112/1//13584_101_1', *);
/* Identification of ISO 13584-101 according to ISO 13584-26 */

/* BSU for component_class */
#50=CLASS_BSU('Bearing', '001', #20);
#60=CLASS_BSU('PAW', '001', #20);
#130=CLASS_BSU('PAW_Geometry', '001', #30);
#140=CLASS_BSU('basic_geometry', '001', #40);

/* BSU for properties */
#90=PROPERTY_BSU('d_in', '001', #50);
#100=PROPERTY_BSU('d_out', '001', #50);
#110=PROPERTY_BSU('e', '001', #50);
#150=PROPERTY_BSU('geometry_level', '001', #140);
#160=PROPERTY_BSU('detail_level', '001', #140);
#170=PROPERTY_BSU('side', '001', #140);
#180=PROPERTY_BSU('prg', '001', #130);
#200=PROPERTY_BSU('variant', '001', #140);
#210=PROPERTY_BSU('unreg_variant', '001', #140);

/* v_c_v range */
#155=VIEW_CONTROL_VARIABLE_RANGE(#150, 1, 1);
#165=VIEW_CONTROL_VARIABLE_RANGE(#160, 2, 2);
#175=VIEW_CONTROL_VARIABLE_RANGE(#170, 1, 6);
#205=VIEW_CONTROL_VARIABLE_RANGE (#200, 1, 1);
#215=VIEW_CONTROL_VARIABLE_RANGE (#210, 0, 0);

/* supplier description */
#31=SUPPLIER_ELEMENT(#30, $, '001', #32, #33);
#32=ORGANIZATION('LISI/ENSMA', 'LISI/ENSMA', '');
#33=ADDRESS($, $, $, $, $, $, $, 'FRANCE', $, $, $, $);

/* Dictionary properties description */
/* prg */
#91=REPRESENTATION_P_DET (#180, $, '001', #92, TEXT('variable used to
reference geometry programs'), $, $, $, $, $, $, $, 'A58', #93, $);

```

```

#92=ITEM_NAMES (LABEL('related program'), (), LABEL(''), $, $);
#93=PROGRAM_REFERENCE_TYPE (
'ISO13584_EXTERNAL_FILE_SCHEMA.PROGRAM_REFERENCE');

/* Dictionary class description */
/* Functional model class view_of_definition*/
#71=FM_CLASS_VIEW_OF(#130, $, '001', #72, TEXT('Explicit functional model
class describing the 2d standard geometry of PAW'), $, $, $, $, (#180),
(), *, *, *, *, *, #140, (#155, #165, #175, #205, #215), (#150, #160, #170,
#200, #210),
(), (), (), #60, (#90, #100, #110), (), (), ());
#72=ITEM_NAMES(LABEL('Functional model class of PAW'), (), LABEL('fm class
of PAW'), $, $);

/* LIBRARY DESCRIPTION */
/* Description of the extension of a functional model
class */
/* It references all the library functional model
instances described themselves by extesion */

#1300=EXPLICIT_FUNCTIONAL_MODEL_CLASS_EXTENSION(#130, (#2501, #2502, #2503,
#2504, #2505, #2506), (#7), (#12), '001', '001', (),
(#90, #170),
(#3000, #3010, #3020, #3030, #3040, #3050,
#3100, #3110, #3120, #3130, #3140, #3150,
#3200, #3210, #3220, #3230, #3240, #3250,
#3300, #3310, #3320, #3330, #3340, #3350,
#3400, #3410, #3420, #3430, #3440, #3450), .T., $, (#90), (), $, $, (), $);

/* Reference to programs which display geometry */
#2501=PROGRAM_REFERENCE(#7, #2601, 'Add1_PAW', 'PAW_p1', (#90, #100, #110),
$, $);
#2502=PROGRAM_REFERENCE(#7, #2602, 'Add2_PAW', 'PAW_p2', (#90, #100, #110),
$, $);
#2503=PROGRAM_REFERENCE(#7, #2603, 'Add3_PAW', 'PAW_p3', (#90, #100, #110),
$, $);
#2504=PROGRAM_REFERENCE(#7, #2604, 'Add4_PAW', 'PAW_p4', (#90, #100, #110),
$, $);
#2505=PROGRAM_REFERENCE(#7, #2605, 'Add5_PAW', 'PAW_p5', (#90, #100, #110),
$, $);
#2506=PROGRAM_REFERENCE(#7, #2606, 'Add6_PAW', 'PAW_p6', (#90, #100, #110),
$, $);
#2601=NOT_TRANSLATABLE_EXTERNAL_CONTENT((#2701));
#2602=NOT_TRANSLATABLE_EXTERNAL_CONTENT((#2702));
#2603=NOT_TRANSLATABLE_EXTERNAL_CONTENT((#2703));
#2604=NOT_TRANSLATABLE_EXTERNAL_CONTENT((#2704));
#2605=NOT_TRANSLATABLE_EXTERNAL_CONTENT((#2705));
#2606=NOT_TRANSLATABLE_EXTERNAL_CONTENT((#2706));
#2701=LANGUAGE_SPECIFIC_CONTENT((#2801), #2801, $);
#2702=LANGUAGE_SPECIFIC_CONTENT((#2802), #2802, $);
#2703=LANGUAGE_SPECIFIC_CONTENT((#2803), #2803, $);
#2704=LANGUAGE_SPECIFIC_CONTENT((#2804), #2804, $);
#2705=LANGUAGE_SPECIFIC_CONTENT((#2805), #2805, $);
#2706=LANGUAGE_SPECIFIC_CONTENT((#2806), #2806, $);

/* Description of the source files for geometry */
#2801=EXTERNAL_FILE_UNIT('PAW_p1.for', '7bit');
#2802=EXTERNAL_FILE_UNIT('PAW_p2.for', '7bit');
#2803=EXTERNAL_FILE_UNIT('PAW_p3.for', '7bit');

```

```

#2804=EXTERNAL_FILE_UNIT('PAW_p4.for', '7bit');
#2805=EXTERNAL_FILE_UNIT('PAW_p5.for', '7bit');
#2806=EXTERNAL_FILE_UNIT('PAW_p6.for', '7bit');

/* GM | FV | FM
*-----*/
/*d_in|side|geometry_level|detail_level|variant |unreg_variant|prg */

/* Description of library functional model instance
by extension */

#3000=LIB_F_MODEL_INSTANCE(#130, (#3001, #3002, #3003, #3004, #3005, #3006,
#3007), ());

/* property values that describe the previous library
functional model extension */

#3001=PROPERTY_VALUE(REAL_VALUE(10), #90);
#3002=PROPERTY_VALUE(1, #170);
#3003=PROPERTY_VALUE(1, #150);
#3004=PROPERTY_VALUE(2, #160);
#3005=PROPERTY_VALUE(1, #200);
#3006=PROPERTY_VALUE(0, #210);
#3007=PROPERTY_VALUE(#2501, #180);

/* Description of library functional model instance
by extension */

#3010=LIB_F_MODEL_INSTANCE(#130, (#3011, #3012, #3013, #3014, #3015, #3016,
#3017), ());

/* property values that describe the previous library
functional model extension */

#3011=PROPERTY_VALUE(REAL_VALUE(10), #90);
#3012=PROPERTY_VALUE(2, #170);
#3013=PROPERTY_VALUE(1, #150);
#3014=PROPERTY_VALUE(2, #160);
#3015=PROPERTY_VALUE(1, #200);
#3016=PROPERTY_VALUE(0, #210);
#3017=PROPERTY_VALUE(#2502, #180);

/* Description of library functional model instance
by extension */

#3020=LIB_F_MODEL_INSTANCE(#130, (#3021, #3022, #3023, #3024, #3025, #3026,
#3027), ());

/* property values that describe the previous library
functional model extension */

#3021=PROPERTY_VALUE(REAL_VALUE(10), #90);
#3022=PROPERTY_VALUE(3, #170);
#3023=PROPERTY_VALUE(1, #150);
#3024=PROPERTY_VALUE(2, #160);
#3025=PROPERTY_VALUE(1, #200);
#3026=PROPERTY_VALUE(0, #210);
#3027=PROPERTY_VALUE(#2503, #180);

```

```

/* Description of library functional model instance
by extension */

#3030=LIB_F_MODEL_INSTANCE(#130, (#3031, #3032, #3033, #3034, #3035, #3036,
#3037), ());

/* property values that describe the previous library
functional model extension */

#3031=PROPERTY_VALUE(REAL_VALUE(10), #90);
#3032=PROPERTY_VALUE(4, #170);
#3033=PROPERTY_VALUE(1, #150);
#3034=PROPERTY_VALUE(2, #160);
#3035=PROPERTY_VALUE(1, #200);
#3036=PROPERTY_VALUE(0, #210);
#3037=PROPERTY_VALUE(#2504, #180);

/* Description of library functional model instance
by extension */

#3040=LIB_F_MODEL_INSTANCE(#130, (#3041, #3042, #3043, #3044, #3045, #3046,
#3047), ());

/* property values that describe the previous library
functional model extension */

#3041=PROPERTY_VALUE(REAL_VALUE(10), #90);
#3042=PROPERTY_VALUE(5, #170);
#3043=PROPERTY_VALUE(1, #150);
#3044=PROPERTY_VALUE(2, #160);
#3045=PROPERTY_VALUE(1, #200);
#3046=PROPERTY_VALUE(0, #210);
#3047=PROPERTY_VALUE(#2505, #180);

/* Description of library functional model instance
by extension */

#3050=LIB_F_MODEL_INSTANCE(#130, (#3051, #3052, #3053, #3054, #3055, #3056,
#3057), ());

/* property values that describe the previous library
functional model extension */

#3051=PROPERTY_VALUE(REAL_VALUE(10), #90);
#3052=PROPERTY_VALUE(6, #170);
#3053=PROPERTY_VALUE(1, #150);
#3054=PROPERTY_VALUE(2, #160);
#3055=PROPERTY_VALUE(1, #200);
#3056=PROPERTY_VALUE(0, #210);
#3057=PROPERTY_VALUE(#2506, #180);

/* Description of library functional model instance
by extension */

#3100=LIB_F_MODEL_INSTANCE(#130, (#3101, #3102, #3103, #3104, #3105, #3106,
#3107), ());

```

```
/* property values that describe the previous library
functional model extension */
```

```
#3101=PROPERTY_VALUE(REAL_VALUE(11), #90);
#3102=PROPERTY_VALUE(1, #170);
#3103=PROPERTY_VALUE(1, #150);
#3104=PROPERTY_VALUE(2, #160);
#3105=PROPERTY_VALUE(1, #200);
#3106=PROPERTY_VALUE(0, #210);
#3107=PROPERTY_VALUE(#2501, #180);
```

```
/* Description of library functional model instance
by extension */
```

```
#3110=LIB_F_MODEL_INSTANCE(#130, (#3111, #3112, #3113, #3114, #3115, #3116,
#3117),());
```

```
/* property values that describe the previous library
functional model extension */
```

```
#3111=PROPERTY_VALUE(REAL_VALUE(11), #90);
#3112=PROPERTY_VALUE(2, #170);
#3113=PROPERTY_VALUE(1, #150);
#3114=PROPERTY_VALUE(2, #160);
#3115=PROPERTY_VALUE(1, #200);
#3116=PROPERTY_VALUE(0, #210);
#3117=PROPERTY_VALUE(#2502, #180);
```

```
/* Description of library functional model instance
by extension */
```

```
#3120=LIB_F_MODEL_INSTANCE(#130, (#3121, #3122, #3123, #3124, #3125, #3126,
#3127),());
```

```
/* property values that describe the previous library
functional model extension */
```

```
#3121=PROPERTY_VALUE(REAL_VALUE(11), #90);
#3122=PROPERTY_VALUE(3, #170);
#3123=PROPERTY_VALUE(1, #150);
#3124=PROPERTY_VALUE(2, #160);
#3125=PROPERTY_VALUE(1, #200);
#3126=PROPERTY_VALUE(0, #210);
#3127=PROPERTY_VALUE(#2503, #180);
```

```
/* Description of library functional model instance
by extension */
```

```
#3130=LIB_F_MODEL_INSTANCE(#130, (#3131, #3132, #3133, #3134, #3135, #3136,
#3137),());
```

```
/* property values that describe the previous library
functional model extension */
```

```
#3131=PROPERTY_VALUE(REAL_VALUE(11), #90);
#3132=PROPERTY_VALUE(4, #170);
```

```

#3133=PROPERTY_VALUE(1, #150);
#3134=PROPERTY_VALUE(2, #160);
#3135=PROPERTY_VALUE(1, #200);
#3136=PROPERTY_VALUE(0, #210);
#3137=PROPERTY_VALUE(#2504, #180);

/* Description of library functional model instance
by extension */

#3140=LIB_F_MODEL_INSTANCE(#130, (#3141, #3142, #3143, #3144, #3145, #3146,
#3147), ());

/* property values that describe the previous library
functional model extension */

#3141=PROPERTY_VALUE(REAL_VALUE(11), #90);
#3142=PROPERTY_VALUE(5, #170);
#3143=PROPERTY_VALUE(1, #150);
#3144=PROPERTY_VALUE(2, #160);
#3145=PROPERTY_VALUE(1, #200);
#3146=PROPERTY_VALUE(0, #210);
#3147=PROPERTY_VALUE(#2505, #180);

/* Description of library functional model instance
by extension */

#3150=LIB_F_MODEL_INSTANCE(#130, (#3151, #3152, #3153, #3154, #3155, #3156,
#3157), ());

/* property values that describe the previous library
functional model extension */

#3151=PROPERTY_VALUE(REAL_VALUE(11), #90);
#3152=PROPERTY_VALUE(6, #170);
#3153=PROPERTY_VALUE(1, #150);
#3154=PROPERTY_VALUE(2, #160);
#3155=PROPERTY_VALUE(1, #200);
#3156=PROPERTY_VALUE(0, #210);
#3157=PROPERTY_VALUE(#2506, #180);

/* Description of library functional model instance
by extension */

#3200=LIB_F_MODEL_INSTANCE(#130, (#3201, #3202, #3203, #3204, #3205, #3206,
#3207), ());

/* property values that describe the previous library
functional model extension */

#3201=PROPERTY_VALUE(REAL_VALUE(13), #90);
#3202=PROPERTY_VALUE(1, #170);
#3203=PROPERTY_VALUE(1, #150);
#3204=PROPERTY_VALUE(2, #160);
#3205=PROPERTY_VALUE(1, #200);
#3206=PROPERTY_VALUE(0, #210);
#3207=PROPERTY_VALUE(#2501, #180);

```

```

/* Description of library functional model instance
by extension */

#3210=LIB_F_MODEL_INSTANCE(#130, (#3211, #3212, #3213, #3214, #3215, #3216,
#3217), ());

/* property values that describe the previous library
functional model extension */

#3211=PROPERTY_VALUE(REAL_VALUE(13), #90);
#3212=PROPERTY_VALUE(2, #170);
#3213=PROPERTY_VALUE(1, #150);
#3214=PROPERTY_VALUE(2, #160);
#3215=PROPERTY_VALUE(1, #200);
#3216=PROPERTY_VALUE(0, #210);
#3217=PROPERTY_VALUE(#2502, #180);

/* Description of library functional model instance
by extension */

#3220=LIB_F_MODEL_INSTANCE(#130, (#3221, #3222, #3223, #3224, #3225, #3226,
#3227), ());

/* property values that describe the previous library
functional model extension */

#3221=PROPERTY_VALUE(REAL_VALUE(13), #90);
#3222=PROPERTY_VALUE(3, #170);
#3223=PROPERTY_VALUE(1, #150);
#3224=PROPERTY_VALUE(2, #160);
#3225=PROPERTY_VALUE(1, #200);
#3226=PROPERTY_VALUE(0, #210);
#3227=PROPERTY_VALUE(#2503, #180);

/* Description of library functional model instance
by extension */

#3230=LIB_F_MODEL_INSTANCE(#130, (#3231, #3232, #3233, #3234, #3235, #3236,
#3237), ());

/* property values that describe the previous library
functional model extension */

#3231=PROPERTY_VALUE(REAL_VALUE(13), #90);
#3232=PROPERTY_VALUE(4, #170);
#3233=PROPERTY_VALUE(1, #150);
#3234=PROPERTY_VALUE(2, #160);
#3235=PROPERTY_VALUE(1, #200);
#3236=PROPERTY_VALUE(0, #210);
#3237=PROPERTY_VALUE(#2504, #180);

/* Description of library functional model instance
by extension */

#3240=LIB_F_MODEL_INSTANCE(#130, (#3241, #3242, #3243, #3244, #3245, #3246,
#3247), ());

```

```

/* property values that describe the previous library
functional model extension */

#3241=PROPERTY_VALUE(REAL_VALUE(13), #90);
#3242=PROPERTY_VALUE(5, #170);
#3243=PROPERTY_VALUE(1, #150);
#3244=PROPERTY_VALUE(2, #160);
#3245=PROPERTY_VALUE(1, #200);
#3246=PROPERTY_VALUE(0, #210);
#3247=PROPERTY_VALUE(#2505, #180);

/* Description of library functional model instance
by extension */

#3250=LIB_F_MODEL_INSTANCE(#130, (#3251, #3252, #3253, #3254, #3255, #3256,
#3257), ());

/* property values that describe the previous library
functional model extension */

#3251=PROPERTY_VALUE(REAL_VALUE(13), #90);
#3252=PROPERTY_VALUE(6, #170);
#3253=PROPERTY_VALUE(1, #150);
#3254=PROPERTY_VALUE(2, #160);
#3255=PROPERTY_VALUE(1, #200);
#3256=PROPERTY_VALUE(0, #210);
#3257=PROPERTY_VALUE(#2506, #180);

/* Description of library functional model instance
by extension */

#3300=LIB_F_MODEL_INSTANCE(#130, (#3301, #3302, #3303, #3304, #3305, #3306,
#3307), ());

/* property values that describe the previous library
functional model extension */

#3301=PROPERTY_VALUE(REAL_VALUE(17), #90);
#3302=PROPERTY_VALUE(1, #170);
#3303=PROPERTY_VALUE(1, #150);
#3304=PROPERTY_VALUE(2, #160);
#3305=PROPERTY_VALUE(1, #200);
#3306=PROPERTY_VALUE(0, #210);
#3307=PROPERTY_VALUE(#2501, #180);

/* Description of library functional model instance
by extension */

#3310=LIB_F_MODEL_INSTANCE(#130, (#3311, #3312, #3313, #3314, #3315, #3316,
#3317), ());

/* property values that describe the previous library
functional model extension */

#3311=PROPERTY_VALUE(REAL_VALUE(17), #90);
#3312=PROPERTY_VALUE(2, #170);
#3313=PROPERTY_VALUE(1, #150);

```

```

#3314=PROPERTY_VALUE(2, #160);
#3315=PROPERTY_VALUE(1, #200);
#3316=PROPERTY_VALUE(0, #210);
#3317=PROPERTY_VALUE(#2502, #180);

/* Description of library functional model instance
by extension */

#3320=LIB_F_MODEL_INSTANCE(#130, (#3321, #3322, #3323, #3324, #3325, #3326,
#3327), ());

/* property values that describe the previous library
functional model extension */

#3321=PROPERTY_VALUE(REAL_VALUE(17), #90);
#3322=PROPERTY_VALUE(3, #170);
#3323=PROPERTY_VALUE(1, #150);
#3324=PROPERTY_VALUE(2, #160);
#3325=PROPERTY_VALUE(1, #200);
#3326=PROPERTY_VALUE(0, #210);
#3327=PROPERTY_VALUE(#2503, #180);

/* Description of library functional model instance
by extension */

#3330=LIB_F_MODEL_INSTANCE(#130, (#3331, #3332, #3333, #3334, #3335, #3336,
#3337), ());

/* property values that describe the previous library
functional model extension */

#3331=PROPERTY_VALUE(REAL_VALUE(17), #90);
#3332=PROPERTY_VALUE(4, #170);
#3333=PROPERTY_VALUE(1, #150);
#3334=PROPERTY_VALUE(2, #160);
#3335=PROPERTY_VALUE(1, #200);
#3336=PROPERTY_VALUE(0, #210);
#3337=PROPERTY_VALUE(#2504, #180);

/* Description of library functional model instance
by extension */

#3340=LIB_F_MODEL_INSTANCE(#130, (#3341, #3342, #3343, #3344, #3345, #3346,
#3347), ());

/* property values that describe the previous library
functional model extension */

#3341=PROPERTY_VALUE(REAL_VALUE(17), #90);
#3342=PROPERTY_VALUE(5, #170);
#3343=PROPERTY_VALUE(1, #150);
#3344=PROPERTY_VALUE(2, #160);
#3345=PROPERTY_VALUE(1, #200);
#3346=PROPERTY_VALUE(0, #210);
#3347=PROPERTY_VALUE(#2505, #180);

/* Description of library functional model instance
by extension */

```

```

#3350=LIB_F_MODEL_INSTANCE(#130, (#3351, #3352, #3353, #3354, #3355, #3356,
#3357), ());

/* property values that describe the previous library
functional model extension */

#3351=PROPERTY_VALUE(REAL_VALUE(17), #90);
#3352=PROPERTY_VALUE(6, #170);
#3353=PROPERTY_VALUE(1, #150);
#3354=PROPERTY_VALUE(2, #160);
#3355=PROPERTY_VALUE(1, #200);
#3356=PROPERTY_VALUE(0, #210);
#3357=PROPERTY_VALUE(#2506, #180);

/* Description of library functional model instance
by extension */

#3400=LIB_F_MODEL_INSTANCE(#130, (#3401, #3402, #3403, #3404, #3405, #3406,
#3407), ());

/* property values that describe the previous library
functional model extension */

#3401=PROPERTY_VALUE(REAL_VALUE(19), #90);
#3402=PROPERTY_VALUE(1, #170);
#3403=PROPERTY_VALUE(1, #150);
#3404=PROPERTY_VALUE(2, #160);
#3405=PROPERTY_VALUE(1, #200);
#3406=PROPERTY_VALUE(0, #210);
#3407=PROPERTY_VALUE(#2501, #180);

/* Description of library functional model instance
by extension */

#3410=LIB_F_MODEL_INSTANCE(#130, (#3411, #3412, #3413, #3414, #3415, #3416,
#3417), ());

/* property values that describe the previous library
functional model extension */

#3411=PROPERTY_VALUE(REAL_VALUE(19), #90);
#3412=PROPERTY_VALUE(2, #170);
#3413=PROPERTY_VALUE(1, #150);
#3414=PROPERTY_VALUE(2, #160);
#3415=PROPERTY_VALUE(1, #200);
#3416=PROPERTY_VALUE(0, #210);
#3417=PROPERTY_VALUE(#2502, #180);

/* Description of library functional model instance
by extension */

#3420=LIB_F_MODEL_INSTANCE(#130, (#3421, #3422, #3423, #3424, #3425, #3426,
#3427), ());

/* property values that describe the previous library
functional model extension */

```

```

#3421=PROPERTY_VALUE(REAL_VALUE(19), #90);
#3422=PROPERTY_VALUE(3, #170);
#3423=PROPERTY_VALUE(1, #150);
#3424=PROPERTY_VALUE(2, #160);
#3425=PROPERTY_VALUE(1, #200);
#3426=PROPERTY_VALUE(0, #210);
#3427=PROPERTY_VALUE(#2503, #180);

/* Description of library functional model instance by extension */

#3430=LIB_F_MODEL_INSTANCE(#130, (#3431, #3432, #3433, #3434, #3435, #3436,
#3437), ());

/* property values that describe the previous library functional model extension */

#3431=PROPERTY_VALUE(REAL_VALUE(19), #90);
#3432=PROPERTY_VALUE(4, #170);
#3433=PROPERTY_VALUE(1, #150);
#3434=PROPERTY_VALUE(2, #160);
#3435=PROPERTY_VALUE(1, #200);
#3436=PROPERTY_VALUE(0, #210);
#3437=PROPERTY_VALUE(#2504, #180);

/* Description of library functional model instance by extension */

#3440=LIB_F_MODEL_INSTANCE(#130, (#3441, #3442, #3443, #3444, #3445, #3446,
#3447), ());

/* property values that describe the previous library functional model extension */

#3441=PROPERTY_VALUE(REAL_VALUE(19), #90);
#3442=PROPERTY_VALUE(5, #170);
#3443=PROPERTY_VALUE(1, #150);
#3444=PROPERTY_VALUE(2, #160);
#3445=PROPERTY_VALUE(1, #200);
#3446=PROPERTY_VALUE(0, #210);
#3447=PROPERTY_VALUE(#2505, #180);

/* Description of library functional model instance by extension */

#3450=LIB_F_MODEL_INSTANCE(#130, (#3451, #3452, #3453, #3454, #3455, #3456,
#3457), ());

/* property values that describe the previous library functional model extension */

#3451=PROPERTY_VALUE(REAL_VALUE(19), #90);
#3452=PROPERTY_VALUE(6, #170);
#3453=PROPERTY_VALUE(1, #150);
#3454=PROPERTY_VALUE(2, #160);
#3455=PROPERTY_VALUE(1, #200);
#3456=PROPERTY_VALUE(0, #210);
#3457=PROPERTY_VALUE(#2506, #180);

```

ENDSEC;
END-ISO-10303-21;

Annex G (informative)
Commented example of library integrated information model 25
physical files.
Exchange of explicit functional models compliant with ISO 13584-
102

G.1. Description of the PAW parts family and its geometry

This annex presents, on an example, an overview of the different resources involved in the description of a parts library using an explicit modeling according to this part of ISO 13584. It describes a physical file example allowing the exchange of explicit functional models compliant with ISO 13584 – 102 view exchange protocol.

This example is similar to the one described in previous annex F. The difference is on the kind of representation to be displayed.

- The example developed in annex F uses geometry representations that are compliant with ISO13584 – 101 view exchange protocol stored in external files (FORTRAN program files) and referenced by the **program_reference** entity.
- The example developed in this annex G uses geometry representations that are compliant with ISO13584 – 102 view exchange defined as STEP representations and referenced by the **representation_reference** entity.

So the only change, regarding annex F, is in the definition of the instances of functional models extension that references **representation_references**..

```

/* LIBRARY DESCRIPTION */
/* Description of the extension of a functional model
class */
/* It references all the library functional model
instances described themselves by extesion */

#1300=EXPLICIT_FUNCTIONAL_MODEL_CLASS_EXTENSION(#130, (
#2501, #2502, #2503, #2504, #2505, #2506,
#2511, #2512, #2513, #2514, #2515, #2516,
#2521, #2522, #2523, #2524, #2525, #2526,
#2531, #2532, #2533, #2534, #2535, #2536,
#2541, #2542, #2543, #2544, #2545, #2546), /* representation references to
files files containing STEP
representation */
(#7), (#12), '001', '001', (), (#90, #170),
(#3000, #3010, #3020, #3030, #3040, #3050,
#3100, #3110, #3120, #3130, #3140, #3150,
#3200, #3210, #3220, #3230, #3240, #3250,
#3300, #3310, #3320, #3330, #3340, #3350,
#3400, #3410, #3420, #3430, #3440, #3450), /* The instances of functional
models described in this
physical file. */
.T., $, (#90), (), $, $ ,(), $);

```

```

/* Enumeration of all the referenced ISO 10303 Step
representation allowing to display geometry */

#2501=REPRESENTATION_REFERENCE(#7, #2601, 'Add1_PAW_front');
...
#2511=REPRESENTATION_REFERENCE(#7, #2611, 'Add2_PAW_front');
...
#2521=REPRESENTATION_REFERENCE(#7, #2621, 'Add3_PAW_front');
...
#2531=REPRESENTATION_REFERENCE(#7, #2631, 'Add4_PAW_front');
...
...
#2541=REPRESENTATION_REFERENCE(#7, #2641, 'Add5_PAW_front');
...

#2601=NOT_TRANSLATABLE_EXTERNAL_CONTENT((#2701));
...
#2611=NOT_TRANSLATABLE_EXTERNAL_CONTENT((#2711));
...
#2621=NOT_TRANSLATABLE_EXTERNAL_CONTENT((#2721));
...
#2631=NOT_TRANSLATABLE_EXTERNAL_CONTENT((#2731));
...
#2641=NOT_TRANSLATABLE_EXTERNAL_CONTENT((#2741));
...
#2701=LANGUAGE_SPECIFIC_CONTENT((#2801), #2801, $);
...
#2711=LANGUAGE_SPECIFIC_CONTENT((#2811), #2811, $);
...
#2721=LANGUAGE_SPECIFIC_CONTENT((#2821), #2821, $);
...
#2731=LANGUAGE_SPECIFIC_CONTENT((#2831), #2831, $);
...
#2741=LANGUAGE_SPECIFIC_CONTENT((#2841), #2841, $);

#2801=EXTERNAL_FILE_UNIT('PAW_p1_front.for', '7bit');
...
#2811=EXTERNAL_FILE_UNIT('PAW_p2_front.for', '7bit');
...
#2821=EXTERNAL_FILE_UNIT('PAW_p3_front.for', '7bit');
...
#2831=EXTERNAL_FILE_UNIT('PAW_p4_front.for', '7bit');
...
#2841=EXTERNAL_FILE_UNIT('PAW_p5_front.for', '7bit');

/*
 * Description of library functional model instance
 * by extension */
#3000=LIB_F_MODEL_INSTANCE(#130, (#3001, #3002, #3003, #3004, #3005, #3006,
#3007), ());

/*
 * property values that describe the previous library
functional model extension */

#3001=PROPERTY_VALUE(REAL_VALUE(10), #90);
#3002=PROPERTY_VALUE(1, #200);
#3003=PROPERTY_VALUE(214, #150);
#3004=PROPERTY_VALUE(2, #160);

```

```
#3005=PROPERTY_VALUE(1, #170);
#3006=PROPERTY_VALUE(1, #210);
#3007=PROPERTY_VALUE(#2501, #180);
...
```

Figure G. 14 — The functional model class extension with reference to STEP representations.

G.2 A complete physical file for explicit functional models compliant with ISO 13584- 102

In this clause, the complete example of a physical file is provided. The following physical file, compliant with LIBRARY INTEGRATED INFORMATION MODEL 25 defines the PAW family and its geometry according to the view exchange protocol defined in ISO 13584 Part 102 which references the **representation** resource defined in ISO 10303 – 42.

```
ISO-10303-21;
HEADER;
FILE_DESCRIPTION(('PLIB EXPLICIT FUNCTIONAL MODEL EXAMPLE 1'), '1');
FILE_NAME('P25_fm_explicit.p21',
          '2000-11-28T16:38:14',
          (''),
          ('LISI/ENSMA'),
          'ECCO RUNTIME SYSTEM BUILT-IN PREPROCESSOR V2.3beta1',
          'ECCO RUNTIME SYSTEM V2.3beta1',
          '');
FILE_SCHEMA('ISO_13584_CC3_CONFORMANCE_SCHEMA');
ENDSEC;

DATA;
#1=GLOBAL_LANGUAGE_ASSIGNMENT('en');

/* Global library description */
#2=LIBRARY_IN_STANDARD_FORMAT(#30, #11, (#7), (#12), (#20, #30,
#40), (#50, #60, #140, #130), (), #3, $, $, ());
#3=ITEM_NAMES(LABEL('Explicit functional model: Geometry'), (),
LABEL('Geometry'), $,
$);
#6=ORGANIZATION('ISO', 'International Organization for Standardization',
 '');
#8=ITEM_NAMES(LABEL('Geometry for Automotive Mechanical Design Processes'),
(),
LABEL('ISO_FDIS_10303_214'), $, $);
#7=STANDARD_DATA_PROTOCOL(#6, $, 'ISO_10303_214', '01', '2', #8, $);
#11=LIBRARY_IIM_IDENTIFICATION('CD', 'ISO13584_25', 2001, '3', $, ());
#12=VIEW_EXCHANGE_PROTOCOL_IDENTIFICATION('DIS', 'ISO13584_102', 2000,
'ISO_10303_214', '2',
(#7), #13);
#13=APPLICATION_PROTOCOL_DEFINITION('FDIS', 'automotive_design_cc02', 2000,
#14);
#14=APPLICATION_CONTEXT('No information available');
#15=APPLICATION_CONTEXT_ELEMENT('Detailed design', #14);

/* DICTIONARY DESCRIPTION */
/*BSU for supplier */
#20=SUPPLIER_BSU('INA', *); /* INA code unknown */
#30=SUPPLIER_BSU('9/19860073600021', *);
/* LISI/ENSMA code in the coding scheme ICD=0009 : SIRET number */
#40=SUPPLIER_BSU('0112/1///13584_102_1', *);
```

```

/* Identification of ISO 13584-102 according to ISO 13584-26 */

/* BSU for component_class */
#50=CLASS_BSU('Bearing', '001', #20);
#60=CLASS_BSU('PAW', '001', #20);
#130=CLASS_BSU('PAW_Geometry', '001', #30);
#140=CLASS_BSU('ISO10303_rep', '001', #40);

/* BSU for properties */
#90=PROPERTY_BSU('d_in', '001', #50);
#100=PROPERTY_BSU('d_out', '001', #50);
#110=PROPERTY_BSU('e', '001', #50);
#150=PROPERTY_BSU('step_ap', '001', #140);
#160=PROPERTY_BSU('step_cc', '001', #140);
#170=PROPERTY_BSU('detail_level', '001', #140);
#180=PROPERTY_BSU('rep', '001', #130);
#200=PROPERTY_BSU('side', '001', #140);
#210=PROPERTY_BSU('variant', '001', #140);

/* v_c_v range */
#155=VIEW_CONTROL_VARIABLE_RANGE(#150, 214, 214);
#165=VIEW_CONTROL_VARIABLE_RANGE(#160, 2, 2);
#175=VIEW_CONTROL_VARIABLE_RANGE(#170, 1, 1);
#205=VIEW_CONTROL_VARIABLE_RANGE (#200, 1, 6);
#215=VIEW_CONTROL_VARIABLE_RANGE (#210, 1, 1);

/* supplier description */
#31=SUPPLIER_ELEMENT(#30, $, '001', #32, #33);
#32=ORGANIZATION('LISI/ENSMA', 'LISI/ENSMA', '');
#33=ADDRESS($, $, $, $, $, $, $, 'FRANCE', $, $, $, $);

/* Dictionary properties description */
/* prg */
#91=REPRESENTATION_P_DET (#180, $, '001', #92, TEXT('variable used to
reference geometry representations'), $, $, $, $, (), $, 'A58', #93, $);
#92=ITEM_NAMES (LABEL('related representation'), (), LABEL(''), $, $);
#93=REPRESENTATION_REFERENCE_TYPE (
'ISO13584_EXTERNAL_FILE_SCHEMA.REPRESENTATION_REFERENCE');

/* Dictionary class description */
/* Functional model class view_of_definition*/
#71=FM_CLASS_VIEW_OF(#130, $, '001', #72, TEXT('Explicit functional model
class describing the 2d geometry of PAW according to STEP AP214'), $, $, $,
$, (#180),
(), *, *, *, *, #140, (#155, #165, #175, #205, #215), (#150, #160, #170,
#200, #210),
(), (), (), #60, (#90, #100, #110), (), (), ());
#72=ITEM_NAMES(LABEL('Functional model class of PAW'), (), LABEL('fm class
of PAW'), $, $);

/* LIBRARY DESCRIPTION */
/* Description of the extension of a functional model
class */
/* It references all the library functional model
instances described themselves by extesnion */

#1300=EXPLICIT_FUNCTIONAL_MODEL_CLASS_EXTENSION(#130, (

```

```

#2501, #2502, #2503, #2504, #2505, #2506,
#2511, #2512, #2513, #2514, #2515, #2516,
#2521, #2522, #2523, #2524, #2525, #2526,
#2531, #2532, #2533, #2534, #2535, #2536,
#2541, #2542, #2543, #2544, #2545, #2546),
(#7), (#12), '001', '001', (), (#90, #170),
(#3000, #3010, #3020, #3030, #3040, #3050,
#3100, #3110, #3120, #3130, #3140, #3150,
#3200, #3210, #3220, #3230, #3240, #3250,
#3300, #3310, #3320, #3330, #3340, #3350,
#3400, #3410, #3420, #3430, #3440, #3450),
.T., $, (#90), (), $, $, (), $);

/* Enumeration of all the referenced ISO 10303 Step
representation allowing to display geometry */

#2501=REPRESENTATION_REFERENCE(#7, #2601, 'Add1_PAW_front');
#2502=REPRESENTATION_REFERENCE(#7, #2602, 'Add1_PAW_rear');
#2503=REPRESENTATION_REFERENCE(#7, #2603, 'Add1_PAW_right');
#2504=REPRESENTATION_REFERENCE(#7, #2604, 'Add1_PAW_left');
#2505=REPRESENTATION_REFERENCE(#7, #2605, 'Add1_PAW_top');
#2506=REPRESENTATION_REFERENCE(#7, #2606, 'Add1_PAW_bottom');

#2511=REPRESENTATION_REFERENCE(#7, #2611, 'Add2_PAW_front');
#2512=REPRESENTATION_REFERENCE(#7, #2612, 'Add2_PAW_rear');
#2513=REPRESENTATION_REFERENCE(#7, #2613, 'Add2_PAW_right');
#2514=REPRESENTATION_REFERENCE(#7, #2614, 'Add2_PAW_left');
#2515=REPRESENTATION_REFERENCE(#7, #2615, 'Add2_PAW_top');
#2516=REPRESENTATION_REFERENCE(#7, #2616, 'Add2_PAW_bottom');

#2521=REPRESENTATION_REFERENCE(#7, #2621, 'Add3_PAW_front');
#2522=REPRESENTATION_REFERENCE(#7, #2622, 'Add3_PAW_rear');
#2523=REPRESENTATION_REFERENCE(#7, #2623, 'Add3_PAW_right');
#2524=REPRESENTATION_REFERENCE(#7, #2624, 'Add3_PAW_left');
#2525=REPRESENTATION_REFERENCE(#7, #2625, 'Add3_PAW_top');
#2526=REPRESENTATION_REFERENCE(#7, #2626, 'Add3_PAW_bottom');

#2531=REPRESENTATION_REFERENCE(#7, #2631, 'Add4_PAW_front');
#2532=REPRESENTATION_REFERENCE(#7, #2632, 'Add4_PAW_rear');
#2533=REPRESENTATION_REFERENCE(#7, #2633, 'Add4_PAW_right');
#2534=REPRESENTATION_REFERENCE(#7, #2634, 'Add4_PAW_left');
#2535=REPRESENTATION_REFERENCE(#7, #2635, 'Add4_PAW_top');
#2536=REPRESENTATION_REFERENCE(#7, #2636, 'Add4_PAW_bottom');

#2541=REPRESENTATION_REFERENCE(#7, #2641, 'Add5_PAW_front');
#2542=REPRESENTATION_REFERENCE(#7, #2642, 'Add5_PAW_rear');
#2543=REPRESENTATION_REFERENCE(#7, #2643, 'Add5_PAW_right');
#2544=REPRESENTATION_REFERENCE(#7, #2644, 'Add5_PAW_left');
#2545=REPRESENTATION_REFERENCE(#7, #2645, 'Add5_PAW_top');
#2546=REPRESENTATION_REFERENCE(#7, #2646, 'Add5_PAW_bottom');

#2601=NOT_TRANSLATABLE_EXTERNAL_CONTENT((#2701));
#2602=NOT_TRANSLATABLE_EXTERNAL_CONTENT((#2702));
#2603=NOT_TRANSLATABLE_EXTERNAL_CONTENT((#2703));
#2604=NOT_TRANSLATABLE_EXTERNAL_CONTENT((#2704));
#2605=NOT_TRANSLATABLE_EXTERNAL_CONTENT((#2705));
#2606=NOT_TRANSLATABLE_EXTERNAL_CONTENT((#2706));

#2611=NOT_TRANSLATABLE_EXTERNAL_CONTENT((#2711));
#2612=NOT_TRANSLATABLE_EXTERNAL_CONTENT((#2712));
#2613=NOT_TRANSLATABLE_EXTERNAL_CONTENT((#2713));
#2614=NOT_TRANSLATABLE_EXTERNAL_CONTENT((#2714));
#2615=NOT_TRANSLATABLE_EXTERNAL_CONTENT((#2715));

```

```

#2616=NOT_TRANSLATABLE_EXTERNAL_CONTENT((#2716));
#2621=NOT_TRANSLATABLE_EXTERNAL_CONTENT((#2721));
#2622=NOT_TRANSLATABLE_EXTERNAL_CONTENT((#2722));
#2623=NOT_TRANSLATABLE_EXTERNAL_CONTENT((#2723));
#2624=NOT_TRANSLATABLE_EXTERNAL_CONTENT((#2724));
#2625=NOT_TRANSLATABLE_EXTERNAL_CONTENT((#2725));
#2626=NOT_TRANSLATABLE_EXTERNAL_CONTENT((#2726));

#2631=NOT_TRANSLATABLE_EXTERNAL_CONTENT((#2731));
#2632=NOT_TRANSLATABLE_EXTERNAL_CONTENT((#2732));
#2633=NOT_TRANSLATABLE_EXTERNAL_CONTENT((#2733));
#2634=NOT_TRANSLATABLE_EXTERNAL_CONTENT((#2734));
#2635=NOT_TRANSLATABLE_EXTERNAL_CONTENT((#2735));
#2636=NOT_TRANSLATABLE_EXTERNAL_CONTENT((#2736));

#2641=NOT_TRANSLATABLE_EXTERNAL_CONTENT((#2741));
#2642=NOT_TRANSLATABLE_EXTERNAL_CONTENT((#2742));
#2643=NOT_TRANSLATABLE_EXTERNAL_CONTENT((#2743));
#2644=NOT_TRANSLATABLE_EXTERNAL_CONTENT((#2744));
#2645=NOT_TRANSLATABLE_EXTERNAL_CONTENT((#2745));
#2646=NOT_TRANSLATABLE_EXTERNAL_CONTENT((#2746));

#2701=LANGUAGE_SPECIFIC_CONTENT((#2801), #2801, $);
#2702=LANGUAGE_SPECIFIC_CONTENT((#2802), #2802, $);
#2703=LANGUAGE_SPECIFIC_CONTENT((#2803), #2803, $);
#2704=LANGUAGE_SPECIFIC_CONTENT((#2804), #2804, $);
#2705=LANGUAGE_SPECIFIC_CONTENT((#2805), #2805, $);
#2706=LANGUAGE_SPECIFIC_CONTENT((#2806), #2806, $);

#2711=LANGUAGE_SPECIFIC_CONTENT((#2811), #2811, $);
#2712=LANGUAGE_SPECIFIC_CONTENT((#2812), #2812, $);
#2713=LANGUAGE_SPECIFIC_CONTENT((#2813), #2813, $);
#2714=LANGUAGE_SPECIFIC_CONTENT((#2814), #2814, $);
#2715=LANGUAGE_SPECIFIC_CONTENT((#2815), #2815, $);
#2716=LANGUAGE_SPECIFIC_CONTENT((#2816), #2816, $);

#2721=LANGUAGE_SPECIFIC_CONTENT((#2821), #2821, $);
#2722=LANGUAGE_SPECIFIC_CONTENT((#2822), #2822, $);
#2723=LANGUAGE_SPECIFIC_CONTENT((#2823), #2823, $);
#2724=LANGUAGE_SPECIFIC_CONTENT((#2824), #2824, $);
#2725=LANGUAGE_SPECIFIC_CONTENT((#2825), #2825, $);
#2726=LANGUAGE_SPECIFIC_CONTENT((#2826), #2826, $);

#2731=LANGUAGE_SPECIFIC_CONTENT((#2831), #2831, $);
#2732=LANGUAGE_SPECIFIC_CONTENT((#2832), #2832, $);
#2733=LANGUAGE_SPECIFIC_CONTENT((#2833), #2833, $);
#2734=LANGUAGE_SPECIFIC_CONTENT((#2834), #2834, $);
#2735=LANGUAGE_SPECIFIC_CONTENT((#2835), #2835, $);
#2736=LANGUAGE_SPECIFIC_CONTENT((#2836), #2836, $);

#2741=LANGUAGE_SPECIFIC_CONTENT((#2841), #2841, $);
#2742=LANGUAGE_SPECIFIC_CONTENT((#2842), #2842, $);
#2743=LANGUAGE_SPECIFIC_CONTENT((#2843), #2843, $);
#2744=LANGUAGE_SPECIFIC_CONTENT((#2844), #2844, $);
#2745=LANGUAGE_SPECIFIC_CONTENT((#2845), #2845, $);
#2746=LANGUAGE_SPECIFIC_CONTENT((#2846), #2846, $);

#2801=EXTERNAL_FILE_UNIT('PAW_p1_front.for', '7bit');
#2802=EXTERNAL_FILE_UNIT('PAW_p1_rear.for', '7bit');
#2803=EXTERNAL_FILE_UNIT('PAW_p1_right.for', '7bit');
#2804=EXTERNAL_FILE_UNIT('PAW_p1_left.for', '7bit');

```

```

#2805=EXTERNAL_FILE_UNIT('PAW_p1_top.for', '7bit');
#2806=EXTERNAL_FILE_UNIT('PAW_p1_bottom.for', '7bit');

#2811=EXTERNAL_FILE_UNIT('PAW_p2_front.for', '7bit');
#2812=EXTERNAL_FILE_UNIT('PAW_p2_rear.for', '7bit');
#2813=EXTERNAL_FILE_UNIT('PAW_p2_right.for', '7bit');
#2814=EXTERNAL_FILE_UNIT('PAW_p2_left.for', '7bit');
#2815=EXTERNAL_FILE_UNIT('PAW_p2_top.for', '7bit');
#2816=EXTERNAL_FILE_UNIT('PAW_p2_bottom.for', '7bit');

#2821=EXTERNAL_FILE_UNIT('PAW_p3_front.for', '7bit');
#2822=EXTERNAL_FILE_UNIT('PAW_p3_rear.for', '7bit');
#2823=EXTERNAL_FILE_UNIT('PAW_p3_right.for', '7bit');
#2824=EXTERNAL_FILE_UNIT('PAW_p3_left.for', '7bit');
#2825=EXTERNAL_FILE_UNIT('PAW_p3_top.for', '7bit');
#2826=EXTERNAL_FILE_UNIT('PAW_p3_bottom.for', '7bit');

#2831=EXTERNAL_FILE_UNIT('PAW_p4_front.for', '7bit');
#2832=EXTERNAL_FILE_UNIT('PAW_p4_rear.for', '7bit');
#2833=EXTERNAL_FILE_UNIT('PAW_p4_right.for', '7bit');
#2834=EXTERNAL_FILE_UNIT('PAW_p4_left.for', '7bit');
#2835=EXTERNAL_FILE_UNIT('PAW_p4_top.for', '7bit');
#2836=EXTERNAL_FILE_UNIT('PAW_p4_bottom.for', '7bit');

#2841=EXTERNAL_FILE_UNIT('PAW_p5_front.for', '7bit');
#2842=EXTERNAL_FILE_UNIT('PAW_p5_rear.for', '7bit');
#2843=EXTERNAL_FILE_UNIT('PAW_p5_right.for', '7bit');
#2844=EXTERNAL_FILE_UNIT('PAW_p5_left.for', '7bit');
#2845=EXTERNAL_FILE_UNIT('PAW_p5_top.for', '7bit');
#2846=EXTERNAL_FILE_UNIT('PAW_p5_bottom.for', '7bit');

/* GM | FV | FM
*-----*/
/*d_in|side|geometry_level|detail_level|variant |unreg_variant|prg */

/* Description of library functional model instance by extension */

#3000=LIB_F_MODEL_INSTANCE(#130, (#3001, #3002, #3003, #3004, #3005, #3006,
#3007), ());

/* property values that describe the previous library functional model extension */

#3001=PROPERTY_VALUE(REAL_VALUE(10), #90);
#3002=PROPERTY_VALUE(1, #200);
#3003=PROPERTY_VALUE(214, #150);
#3004=PROPERTY_VALUE(2, #160);
#3005=PROPERTY_VALUE(1, #170);
#3006=PROPERTY_VALUE(1, #210);
#3007=PROPERTY_VALUE(#2501, #180);

/* Description of library functional model instance by extension */

#3010=LIB_F_MODEL_INSTANCE(#130, (#3011, #3012, #3013, #3014, #3015, #3016,
#3017), ());

/* property values that describe the previous library functional model extension */

```

```

#3011=PROPERTY_VALUE(REAL_VALUE(10), #90);
#3012=PROPERTY_VALUE(2, #200);
#3013=PROPERTY_VALUE(214, #150);
#3014=PROPERTY_VALUE(2, #160);
#3015=PROPERTY_VALUE(1, #170);
#3016=PROPERTY_VALUE(1, #210);
#3017=PROPERTY_VALUE(#2502, #180);

/* Description of library functional model instance
by extension */

#3020=LIB_F_MODEL_INSTANCE(#130, (#3021, #3022, #3023, #3024, #3025, #3026,
#3027), ());

/* property values that describe the previous library
functional model extension */

#3021=PROPERTY_VALUE(REAL_VALUE(10), #90);
#3022=PROPERTY_VALUE(3, #200);
#3023=PROPERTY_VALUE(214, #150);
#3024=PROPERTY_VALUE(2, #160);
#3025=PROPERTY_VALUE(1, #170);
#3026=PROPERTY_VALUE(1, #210);
#3027=PROPERTY_VALUE(#2503, #180);

/* Description of library functional model instance
by extension */

#3030=LIB_F_MODEL_INSTANCE(#130, (#3031, #3032, #3033, #3034, #3035, #3036,
#3037), ());

/* property values that describe the previous library
functional model extension */

#3031=PROPERTY_VALUE(REAL_VALUE(10), #90);
#3032=PROPERTY_VALUE(4, #200);
#3033=PROPERTY_VALUE(214, #150);
#3034=PROPERTY_VALUE(2, #160);
#3035=PROPERTY_VALUE(1, #170);
#3036=PROPERTY_VALUE(1, #210);
#3037=PROPERTY_VALUE(#2504, #180);

/* Description of library functional model instance
by extension */

#3040=LIB_F_MODEL_INSTANCE(#130, (#3041, #3042, #3043, #3044, #3045, #3046,
#3047), ());

/* property values that describe the previous library
functional model extension */

#3041=PROPERTY_VALUE(REAL_VALUE(10), #90);
#3042=PROPERTY_VALUE(5, #200);
#3043=PROPERTY_VALUE(214, #150);
#3044=PROPERTY_VALUE(2, #160);
#3045=PROPERTY_VALUE(1, #170);
#3046=PROPERTY_VALUE(1, #210);
#3047=PROPERTY_VALUE(#2505, #180);

```

```

/* Description of library functional model instance
by extension */

#3050=LIB_F_MODEL_INSTANCE(#130, (#3051, #3052, #3053, #3054, #3055, #3056,
#3057), ());

/* property values that describe the previous library
functional model extension */

#3051=PROPERTY_VALUE(REAL_VALUE(10), #90);
#3052=PROPERTY_VALUE(6, #200);
#3053=PROPERTY_VALUE(214, #150);
#3054=PROPERTY_VALUE(2, #160);
#3055=PROPERTY_VALUE(1, #170);
#3056=PROPERTY_VALUE(1, #210);
#3057=PROPERTY_VALUE(#2506, #180);

/* Description of library functional model instance
by extension */

#3100=LIB_F_MODEL_INSTANCE(#130, (#3101, #3102, #3103, #3104, #3105, #3106,
#3107), ());

/* property values that describe the previous library
functional model extension */

#3101=PROPERTY_VALUE(REAL_VALUE(11), #90);
#3102=PROPERTY_VALUE(1, #200);
#3103=PROPERTY_VALUE(214, #150);
#3104=PROPERTY_VALUE(2, #160);
#3105=PROPERTY_VALUE(1, #170);
#3106=PROPERTY_VALUE(1, #210);
#3107=PROPERTY_VALUE(#2511, #180);

/* Description of library functional model instance
by extension */

#3110=LIB_F_MODEL_INSTANCE(#130, (#3111, #3112, #3113, #3114, #3115, #3116,
#3117), ());

/* property values that describe the previous library
functional model extension */

#3111=PROPERTY_VALUE(REAL_VALUE(11), #90);
#3112=PROPERTY_VALUE(2, #200);
#3113=PROPERTY_VALUE(214, #150);
#3114=PROPERTY_VALUE(2, #160);
#3115=PROPERTY_VALUE(1, #170);
#3116=PROPERTY_VALUE(1, #210);
#3117=PROPERTY_VALUE(#2512, #180);

/* Description of library functional model instance
by extension */

#3120=LIB_F_MODEL_INSTANCE(#130, (#3121, #3122, #3123, #3124, #3125, #3126,
#3127), ());

```

```

/* property values that describe the previous library
functional model extension */

#3121=PROPERTY_VALUE(REAL_VALUE(11), #90);
#3122=PROPERTY_VALUE(3, #200);
#3123=PROPERTY_VALUE(214, #150);
#3124=PROPERTY_VALUE(2, #160);
#3125=PROPERTY_VALUE(1, #170);
#3126=PROPERTY_VALUE(1, #210);
#3127=PROPERTY_VALUE(#2513, #180);

/* Description of library functional model instance
by extension */

#3130=LIB_F_MODEL_INSTANCE(#130, (#3131, #3132, #3133, #3134, #3135, #3136,
#3137), ());

/* property values that describe the previous library
functional model extension */

#3131=PROPERTY_VALUE(REAL_VALUE(11), #90);
#3132=PROPERTY_VALUE(4, #200);
#3133=PROPERTY_VALUE(214, #150);
#3134=PROPERTY_VALUE(2, #160);
#3135=PROPERTY_VALUE(1, #170);
#3136=PROPERTY_VALUE(1, #210);
#3137=PROPERTY_VALUE(#2514, #180);

/* Description of library functional model instance
by extension */

#3140=LIB_F_MODEL_INSTANCE(#130, (#3141, #3142, #3143, #3144, #3145, #3146,
#3147), ());

/* property values that describe the previous library
functional model extension */

#3141=PROPERTY_VALUE(REAL_VALUE(11), #90);
#3142=PROPERTY_VALUE(5, #200);
#3143=PROPERTY_VALUE(214, #150);
#3144=PROPERTY_VALUE(2, #160);
#3145=PROPERTY_VALUE(1, #170);
#3146=PROPERTY_VALUE(1, #210);
#3147=PROPERTY_VALUE(#2515, #180);

/* Description of library functional model instance
by extension */

#3150=LIB_F_MODEL_INSTANCE(#130, (#3151, #3152, #3153, #3154, #3155, #3156,
#3157), ());

/* property values that describe the previous library
functional model extension */

#3151=PROPERTY_VALUE(REAL_VALUE(11), #90);
#3152=PROPERTY_VALUE(6, #200);
#3153=PROPERTY_VALUE(214, #150);

```

```

#3154=PROPERTY_VALUE(2, #160);
#3155=PROPERTY_VALUE(1, #170);
#3156=PROPERTY_VALUE(1, #210);
#3157=PROPERTY_VALUE(#2516, #180);

/* Description of library functional model instance
by extension */

#3200=LIB_F_MODEL_INSTANCE(#130, (#3201, #3202, #3203, #3204, #3205, #3206,
#3207), ());

/* property values that describe the previous library
functional model extension */

#3201=PROPERTY_VALUE(REAL_VALUE(13), #90);
#3202=PROPERTY_VALUE(1, #200);
#3203=PROPERTY_VALUE(214, #150);
#3204=PROPERTY_VALUE(2, #160);
#3205=PROPERTY_VALUE(1, #170);
#3206=PROPERTY_VALUE(1, #210);
#3207=PROPERTY_VALUE(#2521, #180);

/* Description of library functional model instance
by extension */

#3210=LIB_F_MODEL_INSTANCE(#130, (#3211, #3212, #3213, #3214, #3215, #3216,
#3217), ());

/* property values that describe the previous library
functional model extension */

#3211=PROPERTY_VALUE(REAL_VALUE(13), #90);
#3212=PROPERTY_VALUE(2, #200);
#3213=PROPERTY_VALUE(214, #150);
#3214=PROPERTY_VALUE(2, #160);
#3215=PROPERTY_VALUE(1, #170);
#3216=PROPERTY_VALUE(1, #210);
#3217=PROPERTY_VALUE(#2522, #180);

/* Description of library functional model instance
by extension */

#3220=LIB_F_MODEL_INSTANCE(#130, (#3221, #3222, #3223, #3224, #3225, #3226,
#3227), ());

/* property values that describe the previous library
functional model extension */

#3221=PROPERTY_VALUE(REAL_VALUE(13), #90);
#3222=PROPERTY_VALUE(3, #200);
#3223=PROPERTY_VALUE(214, #150);
#3224=PROPERTY_VALUE(2, #160);
#3225=PROPERTY_VALUE(1, #170);
#3226=PROPERTY_VALUE(1, #210);
#3227=PROPERTY_VALUE(#2523, #180);

/* Description of library functional model instance
by extension */

```

```

#3230=LIB_F_MODEL_INSTANCE(#130, (#3231, #3232, #3233, #3234, #3235, #3236,
#3237), ());

/* property values that describe the previous library
functional model extension */

#3231=PROPERTY_VALUE(REAL_VALUE(13), #90);
#3232=PROPERTY_VALUE(4, #200);
#3233=PROPERTY_VALUE(214, #150);
#3234=PROPERTY_VALUE(2, #160);
#3235=PROPERTY_VALUE(1, #170);
#3236=PROPERTY_VALUE(1, #210);
#3237=PROPERTY_VALUE(#2524, #180);

/* Description of library functional model instance
by extension */

#3240=LIB_F_MODEL_INSTANCE(#130, (#3241, #3242, #3243, #3244, #3245, #3246,
#3247), ());

/* property values that describe the previous library
functional model extension */

#3241=PROPERTY_VALUE(REAL_VALUE(13), #90);
#3242=PROPERTY_VALUE(5, #200);
#3243=PROPERTY_VALUE(214, #150);
#3244=PROPERTY_VALUE(2, #160);
#3245=PROPERTY_VALUE(1, #170);
#3246=PROPERTY_VALUE(1, #210);
#3247=PROPERTY_VALUE(#2525, #180);

/* Description of library functional model instance
by extension */

#3250=LIB_F_MODEL_INSTANCE(#130, (#3251, #3252, #3253, #3254, #3255, #3256,
#3257), ());

/* property values that describe the previous library
functional model extension */

#3251=PROPERTY_VALUE(REAL_VALUE(13), #90);
#3252=PROPERTY_VALUE(6, #200);
#3253=PROPERTY_VALUE(214, #150);
#3254=PROPERTY_VALUE(2, #160);
#3255=PROPERTY_VALUE(1, #170);
#3256=PROPERTY_VALUE(1, #210);
#3257=PROPERTY_VALUE(#2526, #180);

/* Description of library functional model instance
by extension */

#3300=LIB_F_MODEL_INSTANCE(#130, (#3301, #3302, #3303, #3304, #3305, #3306,
#3307), ());

/* property values that describe the previous library
functional model extension */

```

```

#3301=PROPERTY_VALUE(REAL_VALUE(17), #90);
#3302=PROPERTY_VALUE(1, #200);
#3303=PROPERTY_VALUE(214, #150);
#3304=PROPERTY_VALUE(2, #160);
#3305=PROPERTY_VALUE(1, #170);
#3306=PROPERTY_VALUE(1, #210);
#3307=PROPERTY_VALUE(#2531, #180);

/* Description of library functional model instance by extension */

#3310=LIB_F_MODEL_INSTANCE(#130, (#3311, #3312, #3313, #3314, #3315, #3316,
#3317), ());

/* property values that describe the previous library functional model extension */

#3311=PROPERTY_VALUE(REAL_VALUE(17), #90);
#3312=PROPERTY_VALUE(2, #200);
#3313=PROPERTY_VALUE(214, #150);
#3314=PROPERTY_VALUE(2, #160);
#3315=PROPERTY_VALUE(1, #170);
#3316=PROPERTY_VALUE(1, #210);
#3317=PROPERTY_VALUE(#2532, #180);

/* Description of library functional model instance by extension */

#3320=LIB_F_MODEL_INSTANCE(#130, (#3321, #3322, #3323, #3324, #3325, #3326,
#3327), ());

/* property values that describe the previous library functional model extension */

#3321=PROPERTY_VALUE(REAL_VALUE(17), #90);
#3322=PROPERTY_VALUE(3, #200);
#3323=PROPERTY_VALUE(214, #150);
#3324=PROPERTY_VALUE(2, #160);
#3325=PROPERTY_VALUE(1, #170);
#3326=PROPERTY_VALUE(1, #210);
#3327=PROPERTY_VALUE(#2533, #180);

/* Description of library functional model instance by extension */

#3330=LIB_F_MODEL_INSTANCE(#130, (#3331, #3332, #3333, #3334, #3335, #3336,
#3337), ());

/* property values that describe the previous library functional model extension */

#3331=PROPERTY_VALUE(REAL_VALUE(17), #90);
#3332=PROPERTY_VALUE(4, #200);
#3333=PROPERTY_VALUE(214, #150);
#3334=PROPERTY_VALUE(2, #160);
#3335=PROPERTY_VALUE(1, #170);
#3336=PROPERTY_VALUE(1, #210);
#3337=PROPERTY_VALUE(#2534, #180);

```

```

/* Description of library functional model instance
by extension */

#3340=LIB_F_MODEL_INSTANCE(#130, (#3341, #3342, #3343, #3344, #3345, #3346,
#3347), ());

/* property values that describe the previous library
functional model extension */

#3341=PROPERTY_VALUE(REAL_VALUE(17), #90);
#3342=PROPERTY_VALUE(5, #200);
#3343=PROPERTY_VALUE(214, #150);
#3344=PROPERTY_VALUE(2, #160);
#3345=PROPERTY_VALUE(1, #170);
#3346=PROPERTY_VALUE(1, #210);
#3347=PROPERTY_VALUE(#2535, #180);

/* Description of library functional model instance
by extension */

#3350=LIB_F_MODEL_INSTANCE(#130, (#3351, #3352, #3353, #3354, #3355, #3356,
#3357), ());

/* property values that describe the previous library
functional model extension */

#3351=PROPERTY_VALUE(REAL_VALUE(17), #90);
#3352=PROPERTY_VALUE(6, #200);
#3353=PROPERTY_VALUE(214, #150);
#3354=PROPERTY_VALUE(2, #160);
#3355=PROPERTY_VALUE(1, #170);
#3356=PROPERTY_VALUE(1, #210);
#3357=PROPERTY_VALUE(#2536, #180);

/* Description of library functional model instance
by extension */

#3400=LIB_F_MODEL_INSTANCE(#130, (#3401, #3402, #3403, #3404, #3405, #3406,
#3407), ());

/* property values that describe the previous library
functional model extension */

#3401=PROPERTY_VALUE(REAL_VALUE(19), #90);
#3402=PROPERTY_VALUE(1, #200);
#3403=PROPERTY_VALUE(214, #150);
#3404=PROPERTY_VALUE(2, #160);
#3405=PROPERTY_VALUE(1, #170);
#3406=PROPERTY_VALUE(1, #210);
#3407=PROPERTY_VALUE(#2541, #180);

/* Description of library functional model instance
by extension */

#3410=LIB_F_MODEL_INSTANCE(#130, (#3411, #3412, #3413, #3414, #3415, #3416,
#3417), ());

```

```

/* property values that describe the previous library
functional model extension */

#3411=PROPERTY_VALUE(REAL_VALUE(19), #90);
#3412=PROPERTY_VALUE(2, #200);
#3413=PROPERTY_VALUE(214, #150);
#3414=PROPERTY_VALUE(2, #160);
#3415=PROPERTY_VALUE(1, #170);
#3416=PROPERTY_VALUE(1, #210);
#3417=PROPERTY_VALUE(#2542, #180);

/* Description of library functional model instance
by extension */

#3420=LIB_F_MODEL_INSTANCE(#130, (#3421, #3422, #3423, #3424, #3425, #3426,
#3427), ());

/* property values that describe the previous library
functional model extension */

#3421=PROPERTY_VALUE(REAL_VALUE(19), #90);
#3422=PROPERTY_VALUE(3, #200);
#3423=PROPERTY_VALUE(214, #150);
#3424=PROPERTY_VALUE(2, #160);
#3425=PROPERTY_VALUE(1, #170);
#3426=PROPERTY_VALUE(1, #210);
#3427=PROPERTY_VALUE(#2543, #180);

/* Description of library functional model instance
by extension */

#3430=LIB_F_MODEL_INSTANCE(#130, (#3431, #3432, #3433, #3434, #3435, #3436,
#3437), ());

/* property values that describe the previous library
functional model extension */

#3431=PROPERTY_VALUE(REAL_VALUE(19), #90);
#3432=PROPERTY_VALUE(4, #200);
#3433=PROPERTY_VALUE(214, #150);
#3434=PROPERTY_VALUE(2, #160);
#3435=PROPERTY_VALUE(1, #170);
#3436=PROPERTY_VALUE(1, #210);
#3437=PROPERTY_VALUE(#2544, #180);

/* Description of library functional model instance
by extension */

#3440=LIB_F_MODEL_INSTANCE(#130, (#3441, #3442, #3443, #3444, #3445, #3446,
#3447), ());

/* property values that describe the previous library
functional model extension */

#3441=PROPERTY_VALUE(REAL_VALUE(19), #90);
#3442=PROPERTY_VALUE(5, #200);
#3443=PROPERTY_VALUE(214, #150);

```

```
#3444=PROPERTY_VALUE(2, #160);
#3445=PROPERTY_VALUE(1, #170);
#3446=PROPERTY_VALUE(1, #210);
#3447=PROPERTY_VALUE(#2545, #180);

/* Description of library functional model instance
by extension */

#3450=LIB_F_MODEL_INSTANCE(#130, (#3451, #3452, #3453, #3454, #3455, #3456,
#3457), ());

/* property values that describe the previous library
functional model extension */

#3451=PROPERTY_VALUE(REAL_VALUE(19), #90);
#3452=PROPERTY_VALUE(6, #200);
#3453=PROPERTY_VALUE(214, #150);
#3454=PROPERTY_VALUE(2, #160);
#3455=PROPERTY_VALUE(1, #170);
#3456=PROPERTY_VALUE(1, #210);
#3457=PROPERTY_VALUE(#2546, #180);

ENDSEC;
END-ISO-10303-21;
```

INDEX

- aggregate_entity_instance_value,15
- aggregate_type,11
- aggregate_value,15
- Allowed_aggregate_values,25
- allowed_reference_to_LIIM_25_rule,55
- applicable property,2
- array_type,13
- array_value,16
- bag_type,12
- bag_value,16
- Basic Semantic Unit,3
- class extension,3
- common dictionary schema,3
- compatible_aggregate_domain_and_value,17
- compatible_aggregate_type_and_value,18
- compatible_complete_types_and_value,24
- compliant_8859_1_protocol_25,57
- compliant_external_file_protocol_25,58
- compliant_http_protocol_25,56
- conformance class,3
- conformance requirements,3
- data element type,3
- data type,3
- DET,3
- dictionary element,3
- entity_instance_type_for_aggregate,10
- family of parts,4
- functional model of a part,4
- functional view of a part,4
- general model of a part,4
- generic family of parts,4
- is_correct_liim_25_application_value,59
- ISO13584_25_conformance_schema,54
- library delivery file,4
- library exchange context,5
- library external file,5
- library integrated information model,5
- library part,4
- library specification of a class,5
- LIIM,5
- list_type,11
- list_value,15
- no_content_without_DET_rule,30, 34, 39, 44, 50
- part,5
- property,5
- representation category,5
- resource construct,6
- set_type,12
- set_value,16
- simple family of parts,6
- supplier library,6
- VEP,6
- view exchange protocol,6
- visible property,6